



Amaravathi
Research Academy

Internet of Things Architecture & Applications



IMRAN KHAN PATAN
MAHABOOB BASHA S
AHMED BASHA SYED

INTERNET OF THINGS - ARCHITECTURE & APPLICATIONS

IMRAN KHAN PATAN
MAHABOOB BASHA S
AHMED BASHA SYED

A TEXT BOOK OF



INTERNET OF THINGS - ARCHITECTURE & APPLICATIONS



By

IMRAN KHAN PATAN

Assistant Professor, Dept of Electronics & Communication Engineering
St.Johns College of Engineering & Technology, Yemmiganur-518360

MAHABOOB BASHA S

Assistant Professor, Dept of Electronics & Communication Engineering
St.Johns College of Engineering & Technology, Yemmiganur-518360

AHMED BASHA SYED

Assistant Professor, Dept of Electronics & Communication Engineering
St.Johns College of Engineering & Technology, Yemmiganur-518360

INTERNET OF THINGS - ARCHITECTURE & APPLICATIONS

Authors: IMRAN KHAN PATAN, MAHABOOB BASHA S, AHMED BASHA SYED



Amaravathi Research Academy.

Plot no. 15, Royal Garden, Kompally Village, MD,
Medchal, Hyderabad, Telangana-500014



All right reserved. No part of this publication may be reproduced or used in any form or by any means - photographic, electronic or mechanical, including photocopying, recording, taping, or information storage and retrieval systems- without the prior written permission of the author.



ISBN: 978-93-6285-228-1

Price: 450/-

09, September 2024



The views expressed by the authors in their articles, reviews etc. in this book are their own. The Editor, Publisher and owner are not responsible for them. All disputes concerning the publication shall be settled in the court at Lunawada.

ABOUT AUTHORS



Imran Khan Patan, working as Assistant Professor, Department of Electronics & Communication Engineering, St.Johns College of Engineering & Technology (Autonomous), Yemmiganur-518360, Kurnool(D), A.P. Currently he is doing his PhD Research in the field of Microwave Communications at JNTUA, Anantapur. He has published 34 Quality papers in a variety of Journals and at National and International Conferences. Additionally, he has provided his skills as a Reviewer to different IEEE, Springer & Scopus Conferences. He also received Best Reviewer Award for his Contribution / Commitment in 9th ICCM-2023, NERIST-Arunachal Pradesh & California State University-USA & France International Association of Academicians (IAASSE), USA.



Mahaboob Basha S, working as Assistant Professor, Department of Electronics & Communication Engineering, St.Johns College of Engineering & Technology (Autonomous), Yemmiganur-518360, Kurnool(D), A.P. He is a Professional Life member for ISTE & IETE Organizations. He is knowledgeable in wide range of subjects. He has Published 21 Quality Papers in various Journals, National/International Conferences. He also provided his skills as a Reviewer to SASI-ITE'24 conference.



Ahmed Basha Syed, working as Assistant Professor, Department of Electronics & Communication Engineering, St.Johns College of Engineering & Technology (Autonomous), Yemmiganur-518360, Kurnool(D), A.P. He is knowledgeable in wide range of subjects. He has Published 21 Quality Papers in various Journals, National/International Conferences. He is a Professional Life member for IEI & IAENG Organizations.

Internet of Things – Architecture & Applications

Imran Khan Patan

**Assistant Professor, St.Johns College of Engineering & Technology,
Yemmiganur-518360, Kurnool(D), A.P. India.
&
Research Scholar, JNTUA, Anantapur, A.P. India.**

Mahaboob Basha S

**Assistant Professor, St.Johns College of Engineering & Technology,
Yemmiganur-518360, Kurnool(D), A.P. India**

Ahmed Basha Syed

**Assistant Professor, St.Johns College of Engineering & Technology,
Yemmiganur-518360, Kurnool(D), A.P. India**

ACKNOWLEDGEMENT

At the very outset we would like to take this opportunity to express our gratitude to almighty for his blessings showered upon us to complete this Book. We are deeply indebted to our parents and family members for their unconditional love and support. Their understanding and encouragement have been a constant source of motivation. They have been on our side during high and low phases of our lives.

This Textbook is the culmination of years of Work, Research, and Collaboration, and it would not have been possible without the support and guidance of many Individuals and Institutions.

We also would like to express our deepest gratitude to our mentors, whose insightful feedback and unwavering support have shaped this book from its inception.

Our thanks also go to the students who have used early drafts of this textbook in their studies and provided feedback that has greatly improved its clarity and effectiveness. Their questions and challenges have pushed us to refine the concepts and ensure that the material is accessible and engaging.

This book is a product of collective effort, and we are profoundly grateful to everyone who has contributed to its creation.

Thank You

IMRAN KHAN PATAN

MAHABOOB BASHA S

AHMED BASHA SYED

PREFACE

The majority of Engineering departments, if not all of them, require students to take a first course in Internet of Things. This book is designed to fulfill that need. The purpose of this is to complement the compulsory text that has been chosen for the course. It offers a condensed presentation of the subject in order to make it simpler for the student to ascertain the primary purpose of each part of the textbook.

Under the framework of a semester system, it is feasible to cover a number of significant areas of both IoT Architecture and IoT Applications. Such adaptability is made possible by this book.

As a matter of fact, there is enough content for a whole Academic year to be covered. In addition to that, we have also supplied Project case Studies using Arduino & Raspberry Pi with a collection of developed projects.

CONTENTS

	Page No.
<i>Acknowledgement</i>	
<i>Preface</i>	
Chapter – 01: Introduction to Internet of Things (IoT)	01-08
Chapter – 02: IoT Applications	09-24
Chapter – 03: Enabling Technologies	25-27
Chapter – 04: Challenges and Barriers of IoT	28-32
Chapter – 05: IoT Reference Architecture	33-44
Chapter – 06: Future of IoT	45-51
Chapter – 07: Arduino for Beginners	52-187
Chapter – 08: Raspberry Pi for Beginners	188-351







Introduction to Internet of Things (IoT)

Now a days, the internet-based information architecture allows the exchange of services and goods between all elements, equipment and objects connected to the network.

The IoT refers to the networked interconnection of those everyday objects, which are often equipped with some kind of intelligence. In this context, Internet can be also a platform for devices to communicate electronically and share information and specific data with the world around them. So, IoT can be seen as a real evolution of what we know as Internet by adding more extensive interconnectivity, a better perception of the information and more comprehensive smart services. For the most part, the Internet was used for connection-oriented application protocols like **HTTP** (*Hypertext Transfer Protocol*) and **SMTP** (*Simple Mail Transfer Protocol*). However, nowadays a large number of smart devices communicate between themselves and to other control systems. This concept is known as **M2M** (*Machine-to-Machine communications*).

IoT (*Internet of things*) is an emerging global Internet-based technical architecture facilitating the exchange of goods and services in global supply chain networks has an impact on the security and privacy of the involved stakeholders.

Some highlights in the IoT history are the following:

- ❖ The term Internet of Things was first used by Kevin Ashton in 1999 that was working in the field of networked **RFID** (*radio frequency identification*) and emerging sensing technologies.
- ❖ However, IoT was “born” sometime between 2008 and 2009.
- ❖ In 2010, the number of everyday physical objects and devices connected to the Internet was around 12.5 billion. Nowadays there are about 25 billion of devices connected to the IoT. More or less a smart device per person.
- ❖ The number of smart devices or “things” connected to the IoT is expected to increase to a further 50 billion by 2020.

The IoT introduces a step change in individuals quality of life by offering a lot of new opportunities to data access, specific services in education, security, health care or transportation among others. On the other hand, it will be a key to increase enterprises productivity by offering a widely distributed, locally intelligent network of smart devices and

new services that can be personalized to customer needs. The IoT brings benefits from improved management and tracking of assets and products, it increases the amount of information data and allows the optimization of equipment and use of resources that can be translated into costs saving. Moreover, it offers the opportunity to create new smart interconnected devices and explore new business models.

Internet of Things (IoT) is the networking of physical objects that contain electronics embedded within their architecture in order to communicate and sense interactions amongst each other or with respect to the external environment. In the upcoming years, IoT-based technology will offer advanced levels of services and practically change the way people lead their daily lives. Advancements in medicine, power, gene therapies, agriculture, smart cities, and smart homes are just a very few of the categorical examples where IoT is strongly established. Over 9 billion 'Things' (physical objects) are currently connected to the Internet, as of now. In the near future, this number is expected to rise to a whopping 20 billion.

1.1. Components used in IoT:

There are four main components used in IoT,

Low-power embedded systems: Less battery consumption and high performance are the inverse factors that play a significant role during the design of electronic systems.

Cloud computing: Data collected through IoT devices is massive and this data has to be stored on a reliable storage server. This is where cloud computing comes into play. The data is processed and learned, giving more room for us to discover where things like electrical faults/errors are within the system.

Availability of big data: We know that IoT relies heavily on sensors, especially in real-time. As these electronic devices spread throughout every field, their usage is going to trigger a massive flux of big data.

Networking connection: In order to communicate, internet connectivity is a must where each physical object is represented by an IP address.

However, there are only a limited number of addresses available according to the IP naming. Due to the growing number of devices, this naming system will not be feasible anymore. Therefore, researchers are looking for another alternative naming system to represent each physical object.

There are two ways of building IoT:

- ❖ Form a separate internet work including only physical objects.
- ❖ Make the Internet ever more expansive, but this requires hard-core technologies such as rigorous cloud computing and rapid big data storage

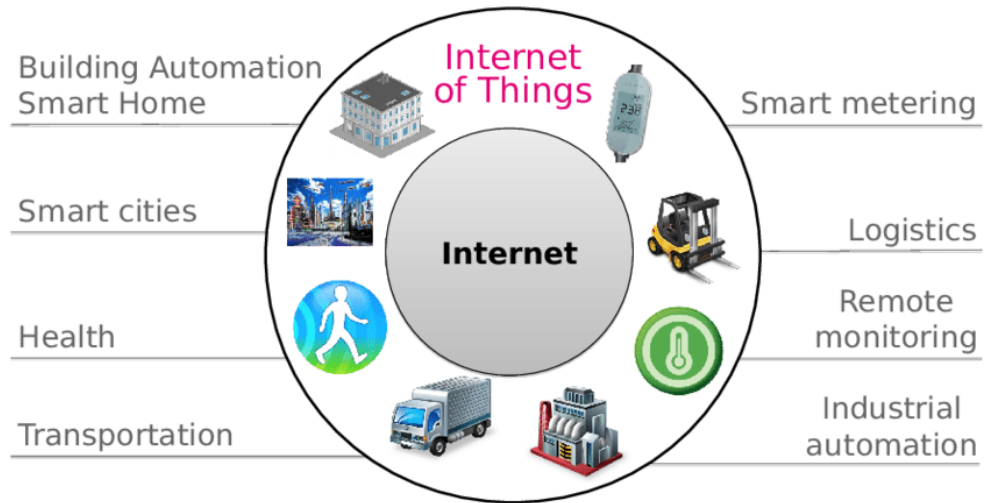


Fig: Internet of Things

Internet of Things:

- ❖ The IoT is a widely used term for a set of technologies, systems, and design principles associated with the emerging wave of Internet-connected things that are based on the physical environment.
- ❖ In many respects, it can initially look the same as M2M communication connecting sensors and other devices to Information and Communication Technology (ICT) systems via wired or wireless networks.
- ❖ In contrast to M2M, however, IoT also refers to the connection of such systems and sensors to the broader Internet, as well as the use of general Internet technologies.
- ❖ In the longer term, it is envisaged that an IoT ecosystem will emerge not dissimilar to today's Internet, allowing things and real world objects to connect, communicate, and interact with one another in the same way humans do via the web today.
- ❖ No longer will the Internet be only about people, media, and content, but it will also

include all real- world assets as intelligent creatures exchanging information, interacting with people, supporting business processes of enterprises, and creating knowledge.

- ❖ The IoT is not a new Internet, it is an extension to the existing Internet. IoT is about the technology, the remote monitoring, and control, and also about where these technologies are applied. IoT can have a focus on the open innovative promises of the technologies at play, and also on advanced and complex processing inside very confined and close environments such as
- ❖ Looking towards the applications and services in the IoT, we see that the application opportunities are open-ended and only imagination will set the limit of what is achievable.
- ❖ Starting from typical M2M applications, one can see application domains emerging that are driven from very diverse needs from across industry, society, and people, and can be of both local interest and global interest.
- ❖ Applications can focus on safety, convenience, or cost reduction, optimizing business processes, or fulfilling various requirements on sustainability and assisted living.
- ❖ Listing all possible application segments is futile, as is providing a ranking of the most important ones. We can point to examples of emerging application domains that are driven by different trends and interests.
- ❖ As can be seen, they are very diverse and can include applications like urban agriculture, robots and food safety tracing, and we will give brief explanations of what these three examples might look like.

1.2. IoT and Digitization:

IoT and digitization are terms that are often used interchangeably. In most contexts, this duality is fine, but there are key differences to be aware of.

At a high level, IoT focuses on connecting “things,” such as objects and machines, to a computer network, such as the Internet. IoT is a well-understood term used across the industry as a whole. On the other hand, digitization can mean different things to different people but generally encompasses the connection of “things” with the data they generate and the business insights that result.

For example, in a shopping mall where Wi-Fi location tracking has been deployed, the “things” are the Wi-Fi devices. Wi-Fi location tracking is simply the capability of

knowing where a consumer is in a retail environment through his or her smart phone's connection to the retailer's Wi-Fi network. While the value of connecting Wi-Fi devices or "things" to the Internet is obvious and appreciated by shoppers, tracking real-time location of

Wi-Fi clients provides a specific business benefit to the mall and shop owners. In this case, it helps the business understand where shoppers tend to congregate and how much time they spend in different parts of a mall or store. Analysis of this data can lead to significant changes to the locations of product displays and advertising, where to place certain types of shops, how much rent to charge, and even where to station security guards.

Digitization, as defined in its simplest form, is the conversion of information into a digital format. Digitization has been happening in one form or another for several decades.

For example, the whole photography industry has been digitized. Pretty much everyone has digital cameras these days, either standalone devices or built in to their mobile phones. Almost no one buys film and takes it to a retailer to get it developed. The digitization of photography has completely changed our experience when it comes to capturing images.

Other examples of digitization include the video rental industry and transportation. In the past, people went to a store to rent or purchase videotapes or DVDs of movies.

With digitization, just about everyone is streaming video content or purchasing movies as downloadable files.

The transportation industry is currently undergoing digitization in the area of taxi services. Businesses such as Uber and Lyft use digital technologies to allow people to get a ride using a mobile phone app. This app identifies the car, the driver, and the fare.

The rider then pays the fare by using the app. This digitization is a major disruptive force to companies providing traditional taxi services.

In the context of IoT, digitization brings together things, data, and business process to make networked connections more relevant and valuable. A good example of this that many people can relate to is in the area of home automation with popular products, such as Nest. With Nest, sensors determine your desired climate settings and also tie in other smart objects, such as smoke alarms, video cameras, and various

third-party devices.

In the past, these devices and the functions they perform were managed and controlled separately and could not provide the holistic experience that is now possible. Nest is just one example of digitization and IoT increasing the relevancy and value of networked, intelligent connections and making a positive impact on our lives.

Companies today look at digitization as a differentiator for their businesses, and IoT is a prime enabler of digitization. Smart objects and increased connectivity drive digitization, and this is one of the main reasons that many companies, countries, and governments are embracing this growing trend.

1.3. IPv6 Introduction:

When we use the Internet for any activity, be it e-mail, data transmission, web browsing, downloading files, images or videos or any other service or application, communication between different network elements and our own computer, laptop or smart phone, uses a protocol: The **IP** (*Internet protocol*) which specifies the technical format of packets and the addressing scheme for computers to communicate over a network.

IPv6 (*Internet protocol version 6*) is the most recent version of the IP, the communications protocol that provides an identification and location system for computers on networks and routes traffic across the Internet.

In order to connect any device to Internet it's necessary to provide an IP address to the device. The first version of an Internet Protocol publicly used was **IPv4** (*Internet protocol version 4*). This protocol was created by the Defense Advanced Research Projects Agency (DARPA). DARPA is an agency of the U.S. Department of Defense responsible for the development of emerging technologies mainly for military applications created in 1958. IPv4 included an addressing system that used numerical identifiers consisting of 32 bits.

The use of addresses with a length of 32 bits limits the total number of possible addresses to a number of approximately 4.3 billion addresses for devices connected to internet around the world. The number of devices connected to Internet will be soon bigger than the number of addresses provided by IPv4. For this reason, and in anticipation of the situation, the agency responsible for standardization of Internet protocols: The **IETF** (*Internet Engineering Task Force*) has been working in a new IP version from 1998: The IPv6, the successor protocol that is intended to replace IPv4 was first formally described in Internet standard document RFC 2460.

IPv6 uses a 128-bit address format, allowing 2^{128} , or approximately 3.4×10^{38} addresses, approximately 8×10^{28} times as many as IPv4. While increasing the pool of addresses is one of the most important benefits of IPv6, there are other important technological changes in IPv6 that will improve the IP protocol: easier administration, better multicast routing, a simpler header format and more efficient routing, built-in authentication and privacy support among others.

IPv6 will coexist with the older IPv4 for some time. The deployment of IPv6 will be made gradually in an orderly coexistence with IPv4. Client devices, network equipment, applications, content and services are to be adapted to the new Internet protocol version IPv6. Moreover, the transition from IPv4 to IPv6 will establish a common set of standards between companies, educational systems, around the world.

IPv6 addresses are represented as eight groups of four hexadecimal digits. These groups are separated by colons, but methods to abbreviate this full notation exist. The IPv6 header format is shown by Fig.

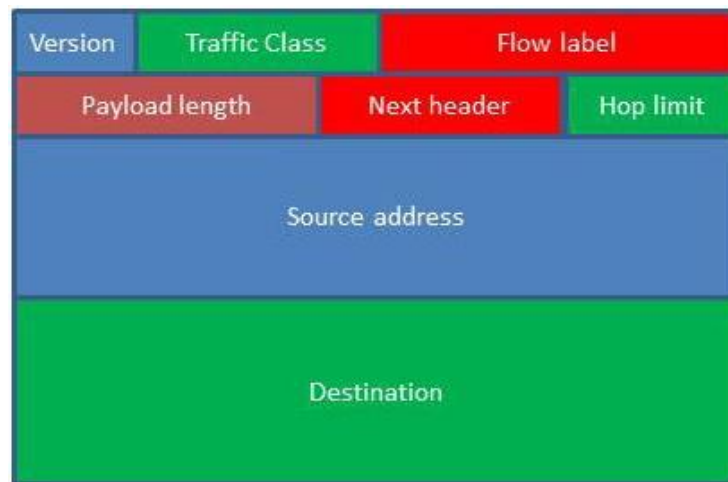


Fig: IPv6 Header Format

The new features introduced with the IPv6 protocol are basically the following:

A new header format, an efficient and hierarchical addressing and routing infrastructure, a much larger address space and stateless and both firewall address configuration, IP security, extensibility, a better Quality of Service (QoS) support and a new protocol for neighboring node interaction.

Structure of IPv6 Header	
Version	4-bit Internet Protocol version number = 6.
Traffic Class	8-bit traffic class field.
Flow Label	20-bit flow label.
Payload Length	16-bit unsigned integer. Length of the IPv6 payload, i.e., the rest of the packet following this IPv6 header, in octets.
Next Header	8-bit selector. Identifies the type of header immediately following the IPv6 header. Uses the same values as the IPv4 protocol field
Hop Limit	8-bit unsigned integer. Decrement by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.
Source Address	128-bit address of the originator of the packet
Destination Address	Address 128-bit address of the intended recipient of the packet (possibly not the ultimate recipient, if a routing header is present).

The IPv6 protocol has solved some of the security problems found in IPv4 networks by adding the **IPsec** (*IP security*) as mandatory. As a result, IPv6 is more efficient.

IPsec enhances the original IP protocol by providing authenticity, integrity, confidentiality and access control to each IP packet through the use of two protocols: **AH** (*authentication header*) and **ESP** (*encapsulating security payload*). Moreover, the expansion of the number of bits in the address field to 128 bits offered by IPv6 creates a significant barrier for attackers wanting to conduct comprehensive port scanning. On the other hand, it is possible to bind a public signature key to an IPv6 address: **CGA** (*Cryptographically Generated Address*).

IPv6 offers also improvements on mobility security. Despite that the MobileIP Internet protocol is available in both IPv4 and IPv6, in IPv6 it was built into the protocol instead of being added as a new function in IPv4. This means that any IPv6 node can use a mobile IP both as required. Mobile IPv6 uses two extensions headline: A routing header for

registration and a headline target to data delivery between mobile nodes and their corresponding fixed nodes.

IoT Applications

IoT Applications:

The IoT can be seen as a combination of sensors and actuators providing and receiving information that is digitalized and placed into bidirectional networks able to transmit all data to be used by a lot of different services and final users.

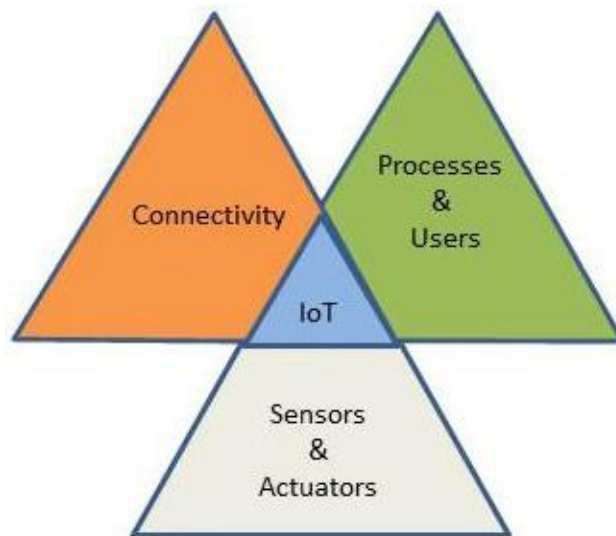


Fig: The IoT Concept.

Multiple sensors can be attached to an object or device in order to measure a broad range of physical variables or phenomena and then transmit all data to the cloud. The sensing can be understood as a service model.

Sensor Classification	
Sensor Data Providers	Business entities that deploy and manage sensors by themselves.
Organizations	Public or Private. Public infrastructures. Commercial organizations. Private corporations: Technology and services providers.
Personal and Households	Mobile phones, smart watches, gyroscopes, cameras, GPS, accelerometers microphones, laptops, food items and household items, such as televisions, cameras, freezers, microwave furnaces, washing machines, smart appliances etc

Now a days, state of the art devices such as conventional house items as refrigerators or televisions comprise communication and sensing capabilities. These capabilities will be constantly increasing by incorporating smarter communication and sensing tools.

Smart connected products capabilities	
Monitoring	The external environment. The product's condition, operations and usage.
Control	Product functions control. Personalization of the user experience. Programming.
Optimization	Predictive diagnostics. Product performance optimization. Costs reduction.
Autonomy	Autonomous product enhancement and personalization. Self-diagnosis and repair. Coordination operation with other products
Efficient decision making process.	Real-time data for decision making.

The architecture of IoT systems can be divided into four layers: Object sensing layer, data exchange layer, information integration layer, and application service layer.

Smart devices can be already connected through traditional Internet. However, the IoT incorporates the sensing layer which reduces the requirements on the capability of those devices and enables the interconnection among them. Sensor data consumers communicate with sensors or sensor owner's through the information integration layer that is responsible of all the communication and transactions. Meanwhile, new requirements and challenges to data exchange, information filtering and integration, definition of new services to users, as well as the complexity of the network architecture. Moreover, the use of cloud technologies is exponentially growing. New infrastructure platforms and software applications are offered in the frame of the IoT.

Some of the major advantages and benefits of the IoT will be the creation of innovative services with improved performance and value added solutions along with the reduction of data acquisition costs of existing services and the opportunity to create new revenue streams in a context of a sustainable business model. These applications can be oriented to consumers, business, commercials, and survey activities, industrial and scientific

community by harnessing the application developers.

Four-layer architecture of IoT	
Object Sensing Layer	Sensing the physical objects and obtaining data.
Data Exchange Layer	Transparent transmission of data through communication networks.
Information Integration Layer	Processing of the uncertain information acquired from the networks, filtering undesired data and integration of main information into usable knowledge for services and final users.
Application Service Layer	Provides content services to users.

2.1. IoT market:

The IoT is an emerging global Internet-based technical architecture facilitating the exchange of goods in a global supply-chain network. As the technology trend shifts towards providing faster data rates and lower latency connectivity the Internet is expected to double in size every 5.3 years and cloud computing can play a key role in that growing. Cloud computing is one of the enabling platforms to support IoT. Most “things” of the real world will be integrated into the virtual world by enabling anytime, anywhere full connectivity.

Cloud computing is a model for enabling access to a shared pool of configurable computing resources by allowing users to take benefit from all existing technologies, without the need for deep knowledge about or expertise with each one of them.

In 2010, the number of everyday physical objects and devices connected to the Internet was around 12.5 billion. This number is expected to double to 25 billion in 2015 as the number of more smart devices per person increases, and to a further 50 billion by 2020.

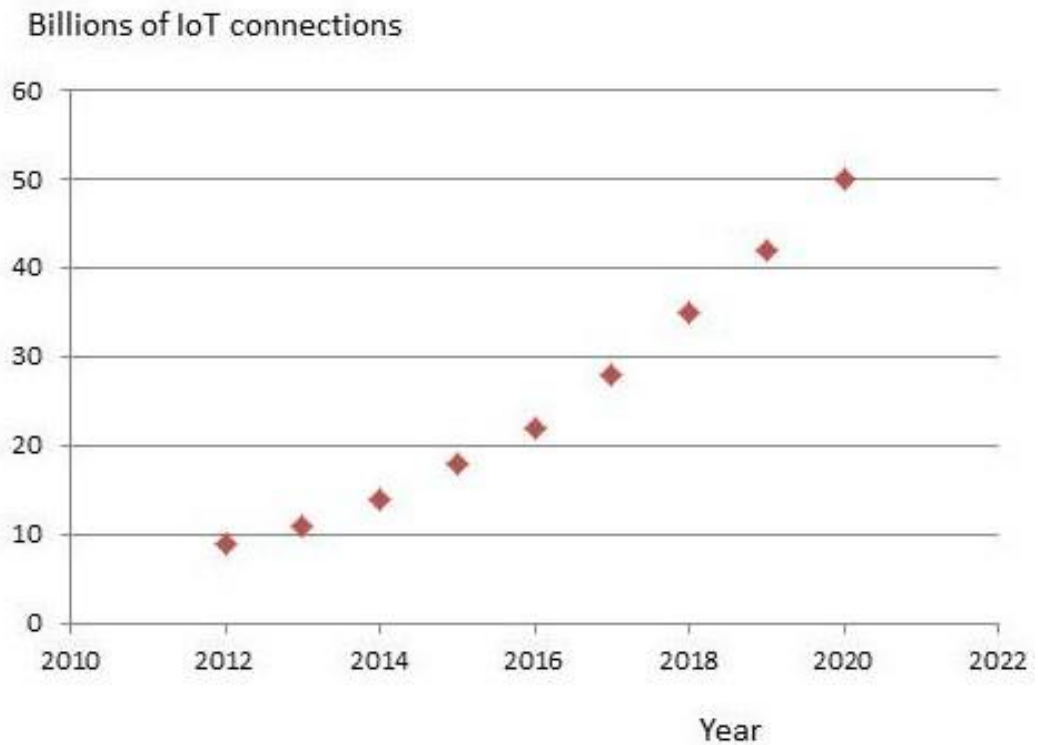


Fig: Number of IoT connections.

Connected World	
31 %	Phones.
29%	Notebook.
10%	Smart Phones.
8%	Smart TV.
5%	Tablets.
5%	Game Players.
5%	Media Players.
5%	eReaders.
3%	Others.

Asia currently has the most M2M connections because of the big effort carried out in some countries as Japan and China. However, American and European technology companies are making an important progress on IoT and they will bring to a market growing in these countries. With the important emergence of the IoT, new regulatory approaches to ensure the privacy and security of users and data must to be defined.

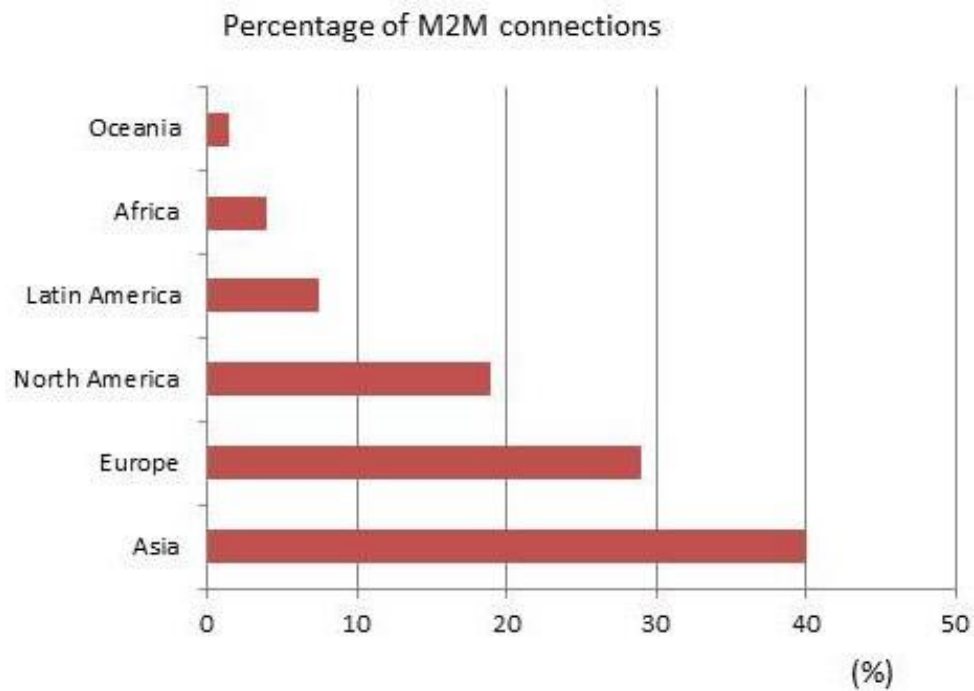


Fig: Percentage of M2M connections.

2.2. Applications:

The number of applications and services that can provide IoT is practically unlimited and can be adapted to many fields of human activity by facilitating and enhancing their quality of life in multiple ways. This chapter gives a short list of applications and services based on IoT. However, it is just a limited description in order to understand all possible new applications and services that IoT could provide. An estimated value about \$19 trillion by 2020 is expected to be achieved by IoT applications and services.

The IoT can find its applications in almost every aspect of our daily life. Below are some of the examples.

Prediction of natural disasters: The combination of sensors and their autonomous coordination and simulation will help to predict the occurrence of land-slides or other natural disasters and to take appropriate actions in advance.

Industry applications: The IoT can find applications in industry e.g., managing a fleet of cars for an organization. The IoT helps to monitor their environmental performance and process the data to determine and pick the one that need maintenance.

Water Scarcity monitoring: The IoT can help to detect the water scarcity at different places. The networks of sensors, tied together with the relevant simulation activities might not only monitor long term water interventions such as catchment area management, but may

even be used to Alert users of a stream, for instance, if an upstream event, such as the accidental release of sewage into the stream, might have dangerous implications.

Design of smart homes: The IoT can help in the design of smart homes e.g., energy consumption management, interaction with appliances, detecting emergencies, home safety and finding things easily, home security etc.

Medical applications: The IoT can also find applications in medical sector for saving lives or improving the quality of life e.g., monitoring health parameters, monitoring activities, support for independent living, The IoT can find its applications in almost every aspect of our daily life. Below are some of the examples.

Prediction of natural disasters: The combination of sensors and their autonomous coordination and simulation will help to predict the occurrence of land-slides or other natural disasters and to take appropriate actions in advance.

Industry applications: The IoT can find applications in industry e.g., managing a fleet of cars for an organization. The IoT helps to monitor their environmental performance and process the data to determine and pick the one that need maintenance.

Water Scarcity monitoring: The IoT can help to detect the water scarcity at different places. The networks of sensors, tied together with the relevant simulation activities might not only monitor long term water interventions such as catchment area management, but may even be used to alert users of a stream, for instance, if an upstream event, such as the accidental release of sewage into the stream, might have dangerous implications.

Design of smart homes: The IoT can help in the design of smart homes e.g., energy consumption management, interaction with appliances, detecting emergencies, home safety and finding things easily, home security etc.

Medical applications: The IoT can also find applications in medical sector for saving lives or improving the quality of life e.g., monitoring health parameters, monitoring activities, support for independent living, monitoring medicines intake etc.

Agriculture application: A network of different sensors can sense data, perform data processing and inform the farmer through communication infrastructure e.g., mobile phone text message about the portion of land that need particular attention. This may include smart packaging of seeds, fertilizer and pest control mechanisms that respond to specific local conditions and indicate actions. Intelligent farming system will help agronomists to have better understanding of the plant growth models and to have efficient farming practices by having the knowledge of land conditions and climate variability. This will significantly

increase the agricultural productivity by avoiding the inappropriate farming conditions.

Intelligent transport system design: The Intelligent transportation system will provide efficient transportation control and management using advanced technology of sensors, information and network. The intelligent transportation can have many interesting features such as non-stop electronic highway toll, mobile emergency command and scheduling, transportation law enforcement, vehicle rules violation monitoring, reducing environmental pollution, anti-theft system, avoiding traffic jams, reporting traffic incidents, smart beaconing, minimizing arrival delays etc.

Design of smart cities: The IoT can help to design smart cities e.g., monitoring air quality, discovering emergency routes, efficient lighting up of the city, watering gardens etc.

Smart metering and monitoring: The IoT design for smart metering and monitoring will help to get accurate automated meter reading and issuance of invoice to the customers. The IoT can be used to design such scheme for wind turbine maintenance and remote monitoring, gas, water as well as environmental metering and monitoring.

Smart Security: The IoT can also find applications in the field of security and surveillance e.g., surveillance of spaces, tracking of people and assets, infrastructure and equipment maintenance, alarming, monitoring medicines intake etc.

Agriculture application: A network of different sensors can sense data, perform data processing and inform the farmer through communication infrastructure e.g., mobile phone text message about the portion of land that need particular attention. This may include smart packaging of seeds, fertilizer and pest control mechanisms that respond to specific local conditions and indicate actions. Intelligent farming system will help agronomists to have better understanding of the plant growth models and to have efficient farming practices by having the knowledge of land conditions and climate variability. This will significantly increase the agricultural productivity by avoiding the inappropriate farming conditions.

2.3. Connected Roadways:

People have been fantasizing about the self-driving car, or autonomous vehicle, in literature and film for decades. While this fantasy is now becoming a reality with well-known projects like Google's self-driving car, IoT is also a necessary component for implementing a fully connected transportation infrastructure. IoT is going to allow self-driving vehicles to better interact with the transportation system around them through bidirectional data exchanges while also providing important data to the riders. Self-driving vehicles need

always-on, reliable communications and data from other transportation-related sensors to reach their full potential. A connected roadway is the term associated with both the driver and driverless cars fully integrating with the surrounding transportation infrastructure. Figure shows a self-driving car designed by Google.



Fig: Self driving car

Basic sensors reside in cars already. They monitor oil pressure, tire pressure, temperature, and other operating conditions, and provide data around the core car functions. From behind the steering wheel, the driver can access this data while also controlling the car using equipment such as a steering wheel, pedals, and so on. The need for all this sensory information and control is obvious. The driver must be able to understand, handle, and make critical decisions while concentrating on driving safely. The Internet of Things is replicating this concept on a much larger scale.

Today, we are seeing automobiles produced with thousands of sensors, to measure everything from fuel consumption to location to the entertainment your family is watching during the ride. As automobile manufacturers strive to re-invent the driving experience, these sensors are becoming IP - enabled to allow easy communication with other systems both inside and outside the car. In addition, new sensors and communication technologies are being developed to allow vehicles to “talk” to other vehicles, traffic signals, school zones, and other elements of the transportation infrastructure. We are now starting to realize a truly connected transportation solution.

2.4. Connected Factory:

For years, traditional factories have been operating at a disadvantage, impeded by production environments that are “disconnected” or, at the very least, “strictly gated” to corporate business systems, supply chains, and customers and partners. Managers of these traditional factories are essentially “flying blind” and lack visibility into their operations. These operations are composed of plant floors, front offices, and suppliers for years, traditional factories have been operating at a disadvantage, impeded by production environments that are “disconnected” or, at the very least, “strictly gated” to corporate business systems, supply chains, and customers and partners. Managers of these traditional factories are essentially “flying blind” and lack visibility into their operations. These operations are composed of plant floors, front offices, and suppliers operating in independent silos. Consequently, rectifying downtime issues, quality problems, and the root causes of various manufacturing inefficiencies is often difficult.

The main challenges facing manufacturing in a factory environment today include the following:

- ✓ Accelerating new product and service introductions to meet customer and market opportunities
- ✓ Increasing plant production, quality, and uptime while decreasing cost
- ✓ Mitigating unplanned downtime (which wastes, on average, at least 5% of production)
- ✓ Securing factories from cyber threats
- ✓ Decreasing high cabling and re-cabling costs (upto 60% of deployment costs)
- ✓ Improving worker productivity and safety

Adding another level of complication to these challenges is the fact that they often need to be addressed at various levels of the manufacturing business. For example, executive management is looking for new ways to manufacture in a more cost-effective manner while balancing the rising energy and material costs. Product development has time to market as the top priority. Plant managers are entirely focused on gains in plant efficiency and operational agility. The controls and automation department looks after the plant networks, controls, and applications and therefore requires complete visibility into all these systems.

Industrial enterprises around the world are retooling their factories with advanced technologies and architectures to resolve these problems and boost manufacturing flexibility and speed. These improvements help them achieve new levels of overall equipment effectiveness, supply chain responsiveness, and customer satisfaction. A convergence of

factory-based operational technologies and architectures with global IT networks is starting to occur, and this is referred to as the connected factory.

As with the IoT solutions for the connected roadways previously discussed, there are already large numbers of basic sensors on factory floors. However, with IoT, these sensors not only become more advanced but also attain a new level of connectivity. They are smarter and gain the ability to communicate, mainly using the Internet Protocol (IP) over an Ethernet infrastructure.

In addition to sensors, the devices on the plant floor are becoming smarter in their ability to transmit and receive large quantities of real-time informational and diagnostic data. Ethernet connectivity is becoming pervasive and spreading beyond just the main controllers in a factory to devices such as the robots on the plant floor. In addition, more IP-enabled devices, including video cameras, diagnostic smart objects, and even personal mobile devices, are being added to the manufacturing environment.

For example, a smelting facility extracts metals from their ores. The facility uses both heat and chemicals to decompose the ore, leaving behind the base metal. This is a multistage process, and the data and controls are all accessed via various control rooms in a facility. Operators must go to a control room that is often hundreds of meters away for data and production changes. Hours of operator time are often lost to the multiple trips to the control room needed during a shift. With IoT and a connected factory solution, true “machine-to-people” connections are implemented to bring sensor data directly to operators on the floor via mobile devices. Time is no longer wasted moving back and forth between the control rooms and the plant floor. In addition, because the operators now receive data in real time, decisions can be made immediately to improve production and fix any quality problems.

Another example of a connected factory solution involves a real-time location system (RTLS). An RTLS utilizes small and easily deployed Wi-Fi RFID tags that attach to virtually any material and provide real-time location and status. These tags enable a facility to track production as it happens. These IoT sensors allow components and materials on an assembly line to “talk” to the network. If each assembly line’s output is tracked in real time, decisions can be made to speed up or slow production to meet targets, and it is easy to determine how quickly employees are completing the various stages of production. Bottlenecks at any point in production and quality problems are also quickly identified.

2.5. Smart Connected Buildings:

Another place IoT is making a disruptive impact is in the smart connected buildings space. In the past several decades, buildings have become increasingly complex, with systems overlaid one upon another, resulting in complex intersections of structural, mechanical, electrical, and IT components. Over time, these operational networks that support the building environment have matured into sophisticated systems; however, for the most part, they are deployed and managed as separate systems that have little to no interaction with each other.

The function of a building is to provide a work environment that keeps the workers comfortable, efficient, and safe. Work areas need to be well lit and kept at a comfortable temperature. To keep workers safe, the fire alarm and suppression system needs to be carefully managed, as do the door and physical security alarm systems. While intelligent systems for modern buildings are being deployed and improved for each of these functions, most of these systems currently run independently of each other—and they rarely take into account where the occupants of the building actually are and how many of them are present in different parts of the building.

However, many buildings are beginning to deploy sensors throughout the building to detect occupancy. These tend to be motion sensors or sensors tied to video cameras. Motion detection occupancy sensors work great if everyone is moving around in a crowded room and can automatically shut the lights off when everyone has left, but what if a person in the room is out of sight of the sensor. It is a frustrating matter to be at the mercy of an unintelligent sensor on the wall that wants to turn off the lights on you.

Similarly, sensors are often used to control the heating, ventilation, and air-conditioning (HVAC) system. Temperature sensors are spread throughout the building and are used to influence the building management system's (BMS's) control of air flow into a room.

Another interesting aspect of the smart building is that it makes them easier and cheaper to manage. Considering the massive costs involved in operating such complex structures, not to mention how many people spend their working lives inside a building, managers have become increasingly interested in ways to make buildings more efficient and cheaper to manage. Have you ever heard people complain that they had too little working space in their office, or that the office space wasn't being used efficiently? When people go to their managers and ask for a change to the floor plan, such as asking for an increase in the amount of space they work in, they are often asked to prove their case. But workplace floor

efficiency and usage evidence tend to be anecdotal at best. When smart building sensors and occupancy detection are combined with the power of data analytics it becomes easy to demonstrate floor plan usage and prove your case. Alternatively, the building manager can use a similar approach to see where the floor is not being used efficiently and use this information to optimize the available space. This has brought about the age of building automation, empowered by IoT. Another promising IoT technology in the smart connected building, and one that is seeing wide spread adoption, is the “digital ceiling.” The digital ceiling is more than just a lighting control system. This technology encompasses several of the building’s different networks—including lighting, HVAC, blinds, CCTV (closed-circuit television), and security systems—and combines them into a single IP network.

2.6. Smart Creatures:

When you think about IoT, you probably picture only inanimate objects and machines being connected. However, IoT also provides the ability to connect living things to the Internet. Sensors can be placed on animals and even insects just as easily as on machines, and the benefits can be just as impressive.

One of the most well-known applications of IoT with respect to animals focuses on what is often referred to as the “connected cow.” Sparked, a Dutch company, developed a sensor that is placed in a cow’s ear. The sensor monitors various health aspects of the cow as well as its location and transmits the data wirelessly for analysis by the farmer.

The data from each of these sensors is approximately 200 MB per year, and you obviously need a network infrastructure to make the connection with the sensors and store the information. Once the data is being collected, however, you get a complete view of the herd, with statistics on every cow. You can learn how environmental factors may be affecting the herd as a whole and about changes in diet. This enables early detection of disease as cows tend to eat less days before they show symptoms. These sensors even allow the detection of pregnancy in cows.

Another application of IoT to organisms involves the placement of sensors on roaches. While the topic of roaches is a little unsettling to many folks, the potential benefits of IoT-enabled roaches could make a life-saving difference in disaster situations.

2.7. Convergence of IT and OT:

Until recently, information technology (IT) and operational technology (OT) have for the most part lived in separate worlds. IT supports connections to the Internet along with related data and technology systems and is focused on the secure flow of data across an organization. OT monitors and controls devices and processes on

physical operational systems. These systems include assembly lines, utility distribution networks, production facilities, roadway systems, and many more. Typically, IT did not get involved with the production and logistics of OT environments.

Specifically, the IT organization is responsible for the information systems of a business, such as email, file and print services, databases, and so on. In comparison, OT is responsible for the devices and processes acting on industrial equipment, such as factory machines, meters, actuators, electrical distribution automation devices, SCADA (supervisory control and data acquisition) systems, and so on. Traditionally, OT has used dedicated networks with specialized communications protocols to connect these devices, and these networks have run completely separately from the IT networks.

Management of OT is tied to the lifeblood of a company. For example, if the network connecting the machines in a factory fails, the machines cannot function, and production may come to a stand still, negatively impacting business on the order of millions of dollars. On the other hand, if the email server (run by the IT department) fails for a few hours, it may irritate people, but it is unlikely to impact business at anywhere near the same level. Table highlights some of the differences between IT and OT networks and their various challenges.

With the rise of IoT and standards-based protocols, such as IPv6, the IT and OT worlds are converging or, more accurately, OT is beginning to adopt the network protocols, technology, transport, and methods of the IT organization, and the IT organization is beginning to support the operational requirements used by OT. When IT and OT begin using the same networks, protocols, and processes, there are clear economies of scale. Not only does convergence reduce the amount of capital infrastructure needed but networks become easier to operate, and the flexibility of open standards allows faster growth and adaptability to new technologies.

With the merging of OT and IT, improvements are being made to both systems. OT is looking more toward IT technologies with open standards, such as Ethernet and IP. At the same time, IT is becoming more of a business partner with OT by better understanding business outcomes and operational requirements.

The overall benefit of IT and OT working together is a more efficient and profitable business due to reduced down time, lower costs through economy of scale, reduced inventory, and improved delivery times. When IT/OT convergence is managed correctly, IoT becomes fully supported by both groups. This provides a “best of both worlds” scenario, where solid industrial control systems reside on an open, integrated, and secure technology foundation.

2.8. IoT Applications and services:

- ❖ Connected intelligent buildings: Improvements in efficiency (energy management and saving) and security (sensors and alarms). Domestic applications including smart sensors and actuators to control home appliances. Health and education services at home. Remote control of treatments for patients. Cable/satellite services. Energy storage/generation systems. Automatic shutdown of electronics when not in use. Smart thermostats. Smoke detectors and alarms. Access control applications. Smart door locks. Sensors built into building infrastructure to guide first responders and assistances. Safety for all family members.
- ❖ Smart cities and transportation: Integration of security services. Optimization of public and private transportation. Parking Sensors. Smart management of parking services and traffic in real time. Smart management of traffic lights depending on traffic queues. Locate cars that have over stayed Smart energy grids. Security (cameras, smart sensors, information to citizens). Water management. Parks and Gardens irrigation. Smart garbage cans. Pollution and mobility controls. Get immediate feedback and opinions from citizens. Smart governance. Voting Systems. Accident monitoring, emergency actions coordination.



Fig: Example of IoT applications: Smart cities.

- ❖ Education: Linking virtual and physical classrooms to make learning more efficient and accessible, e-learning. Access services to virtual libraries and educational portals. Interchange of reports and results in real time. Lifelong learning. Foreign languages learning. Attendance management.
- ❖ Consumer electronics: Smart phones. Smart TV. Laptops, computers and tablets. Smart refrigerators, washers and dryers. Smart home theatre systems. Smart appliances. Pet collar sensors. Personalization of the user experience. Autonomous product operation. Personal locators. Smart glasses.
- ❖ Health: Monitoring of chronic diseases. Improvement of the quality of care and quality of life for patients. Activity Trackers. Remote diagnostic. Connected bracelets. Interactive belts. Sport and fitness monitoring. Intelligent tags for drugs. Drug usage tracking. Biochips. Brain-computer interfaces. Monitoring eating habits.
- ❖ Automotive: Smart Cars. Traffic control. Advance information about what is broken. Wireless monitoring of tire pressure of car. Smart energy management and control. Self-diagnosis. Accelerometers. Position, presence and proximity sensors. Analysis of the best way to go in real time. GPS tracking. Vehicle speed control. Autonomous vehicles using IoT services.
- ❖ Agriculture and environment: Measurement and monitoring of environmental pollution (CO₂, noise, contaminant elements presents in ambient). Forecasting climate changes based on smart sensors monitoring. Passive RFID tags attached to agriculture products. Sensors in pallets of products. Waste management. Nutrition calculations.
- ❖ Energy services: accurate data on energy consumption. Smart metering. Smart grids. Analysis and prediction of energy consumption behaviors and patterns. Forecasting future energy trends and needs. Wireless sensors networks. Energy harvesting and recycling.
- ❖ Smart Connectivity: Data management and service provisioning. Use of social media and social networking. Access to email, voice and video services, Interactive group communication, Real time streaming, Interactive gaming, Augmented reality, Network security monitoring, Wearable user interfaces, Affective computing, Biometric authentication methods, Consumer telemetric, M2M communication services, Big data analysis, Virtual reality, Cloud computing services, Ubiquitous

computing, Computer vision, Smart antennas.

- ❖ Manufacturing: Gas and flow sensors, Smart sensors of humidity, temperature, motion, force, load, leaks/levels. Machine vision, Acoustic and vibration sensing, Compound applications, Smart control of robots, Control and optimization of fabrication processes, Pattern recognition, Machine Learning, Predictive Analytics. Mobile logistics, Warehouse management, Prevent overproduction, Efficient logistics.
- ❖ Shopping: Intelligent shopping, RFID and other electronic tags and readers, Barcodes in retail, Inventory control, Control of geographical origin of food and products, Control food quality and safety.

Enabling Technologies

Successful application of the IoT concept into the real world is possible thanks to advancements in underlying technologies. In this section the most relevant enabling technologies will be stated with the aim to provide a picture of the role they will likely play in the IoT.

Energy:

Power and energy storage technologies are enablers for the deployment of IoT applications. Energy issues, in all its phases, from harvesting to conservation and usage, are central to the development of the IoT. These technologies have to provide high power-density energy generation and harvesting solutions which, when used with today's low power nano electronics, will enable us to design self-powered intelligent sensor-based wireless identifiable device. There is still a need to research and develop solutions in this area (nano electronics, semiconductor, sensor technology, micro systems integration) having as an objective ultralow power devices, and more efficient and compact energy storage like batteries, fuel cells, and printed/polymer batteries, as current devices seem inadequate considering the processing power needed and energy limitations of the future. In addition, system integration will increase efficiency of current systems, and will provide a number of solutions for the future needs.

Sensors:

Sensors are one of the key building blocks of the Internet of Things. As ubiquitous systems, they can be deployed everywhere. They can also be implanted under human skin, in a purse or on a T-shirt. Some can be as small as four millimetres in size, but the data they collect can be received hundreds of miles away. They complement human senses and have become indispensable in a large number of industries, from health care to construction. Sensors have the key advantage that they can anticipate human needs based on information collected about their context. Their intelligence multiplied by numerous networks allows them not only to report about external environment, but also to take action without human intervention.

Miniaturized silicon chips are designed with new capabilities in smaller form factors and better processing performance and efficiency, Costs are falling, following the Moore's Law, The cost of bandwidth has also declined and similarly the processing costs, enabling

more devices to be not just connected, but smart enough to know what to do with all the new data they are generating or receiving.

Capabilities such as context awareness and inter-machine communication are considered a high priority for the IoT. Additional priorities are the integration of memory and processing power, the capacity of resisting harsh environments, and an affordable security. Furthermore, the development of ultralow power processors/microcontrollers cores designed specifically for mobile IoT devices and a new class of simple and affordable IoT-centric smart systems will be an enabling factor.

The solutions in this respect will range from micro programmed finite state machines to the use of microcontrollers. The choice is a trade-off between flexibility, programmability, silicon area, and power consumption. The devices require some form of non-volatile storage (EEPROM/FRAM/Polymer), independent of whether this will be laser trimmed at the time of manufacture, one time programmable, or electrically rewritable. Rewritable non-volatile memory is clearly preferred for achieving high throughput during production test, and allows concurrently the benefit of user memory, programmability and storage of sensor data.

Cloud Computing:

Cloud computing is a model for on-demand access to a shared pool of configurable resources (e.g., computers, networks, servers, storage, applications, services, software) that can be provisioned as Infrastructure as a Service (IaaS) or Software as a Service (SaaS). One of the most important outcomes of the IoT is an enormous amount of data generated from devices connected to the Internet. Many IoT applications require massive data storage, huge processing speed to enable real time decision making, and high-speed broadband networks to stream data, audio, or video. Cloud computing provides an ideal back-end solution for handling huge datastreams and processing them for the unprecedented number of IoT devices and humans in real time.

Communication:

New, smart multi frequency band antennas, integrated on-chip and made of new materials are the communication means that will enable the devices to communicate. On-chip antennas must be optimized for size, cost and efficiency, and could come in various forms like coil on chip, printed antennas, embedded antennas, and multiple antenna using different substrates and 3D structures. Modulation schemes and transmission speed are also important issues to be tackled allowing multi-frequency energy efficient communication protocols and

transmission rates. The communication protocols will be designed for Web oriented architectures of the IoT platform where all objects, wireless devices, cameras, PCs etc. are combined to analyze location, intent and even emotions over a network. New methods of effectively managing power consumption at different levels of the network design are needed, from network routing down to the architecture of individual devices.

Integration:

Integration of smart devices into packaging, or better, into the products themselves will allow a significant cost saving and increase the Eco friendliness of products. The use of integration of chips and antennas into non-standard substrates like textiles and paper, and the development of new substrates, conducting paths and bonding materials adequate for harsh environments and for ecologically sound disposal will continue. System-in-Package (SiP) technology allows flexible and 3D integration of different elements such as antennas, sensors, active and passive components into the packaging, improving performance and reducing the tag cost. RFID inlays with a strap coupling structure are used to connect the integrated circuit chip and antenna in order to produce a variety of shapes and sizes of labels, instead of direct mounting.

Standards:

IoT devices are quite diverse and measure different parameters and with different conventions and units of measure. Though competing proprietary protocols keep getting proposed, it is likely that open source standards will be one of the ways to get this data to interoperate.

Clearly, open standards are key enablers for the success of wireless communication technologies and, in general, for any kind of Machine-to-Machine communication. However, the need for faster setting of interoperable standards has been recognized an important element for IoT applications deployment. Clarification on the requirements for a unique global identification, naming and resolver is needed. Lack of convergence of the definition of common reference models, reference architecture for the Future Networks, Future Internet and IoT and integration of legacy systems and networks is a challenge that has to be addressed in the future.

Challenges and barriers of IoT

Many challenging issues still need to be addressed. Addressing these challenges enables service providers and application programmers to implement their services efficiently. In the following paragraphs, we provide a brief discussion of the main challenges faced in the development and deployment phases of the IoT.

Reliability:

Reliability aims to increase the success rate of IoT service delivery. It has a close relationship with availability as by reliability, we guarantee the availability of information and services over time. Reliability is even more critical and has more stringent requirements when it comes to the field of emergency response applications. In these systems, the critical part is the communication network which must be resilient to failures in order to realize reliable information distribution. Reliability must be implemented in software and hardware throughout all the IoT layers. In order to have an efficient IoT, the underlying communication must be reliable, because for example by an unreliable perception, data gathering, processing, and transmission can lead to long delays, loss of data, and eventually wrong decisions, which can lead to disastrous scenarios and can consequently make the IoT less dependable.

Reliability refers to the proper working of the system based on its specification.

Performance:

Evaluating the performance of IoT services is a big challenge since it depends on the performance of many components as well as the performance of the underlying technologies. The IoT, like other systems, needs to continuously develop and improve its services to meet requirements of customers. The IoT devices need to be monitored and evaluated to provide the best possible performance at an affordable price for customers. Many metrics can be used to assess the performance of the IoT including the processing speed, communication speed, device form factor, and cost.

Performance evaluation of the individual underlying protocols and technologies, application layer protocols, and QoS have been reported in the literature, but the lack of a thorough performance evaluation for IoT applications is still an open issue.

Quality of service (QoS) is the overall performance of a telephony or computer network, particularly the performance seen by the users of the network.

Interoperability:

End-to-end interoperability is another challenge for the IoT due to the need to handle a large number of heterogeneous things that belong to different platforms. Interoperability should be considered by both application developers and IoT device manufactures to ensure the delivery of services for all customers regardless of the specifications of the hardware platform that they use. For example, most of the smart phones nowadays support common communication technologies such as Wi-Fi, NFC, and GSM to guarantee the interoperability in different scenarios.

Also, programmers of the IoT should build their applications to allow for adding new functions without causing problems or losing functions while maintaining integration with different communication technologies. Consequently, interoperability is a significant criterion in designing and building IoT services to meet requirements of customers. Beside variety of protocols, different interpretations of the same standard implemented by different parties presents a challenge for interoperability. To avoid such ambiguities, interoperability testing between different products in a test bed like ETSI Plug tests would be helpful. PROBE-IT is a research project that aims to ensure the interoperability of validated IoT solutions that conducted interoperability tests like CoAP, 6LoWPAN, and IoT semantic interoperability.

It is a known fact that two different devices might not be interoperable, even if they are following the same standard. This is a major show stopper for wide adoption of IoT technologies. Future tags must integrate different communication standards and protocols that operate at different frequencies and allow different architectures, centralized or distributed, and be able to communicate with other networks unless global, well defined standards emerge.

Security and Privacy:

Security presents a significant challenge for the IoT implementations due to the lack of common standard and architecture for the IoT security. In heterogeneous networks as in the case of the IoT, it is not easy to guarantee the security and privacy of users. The core functionality of the IoT is based on the exchange of information between billions or even trillions of Internet connection objects. One open problem in IoT security that has not been considered in the standards is the distribution of the keys amongst devices. On the other hand,

privacy issues and profile access operations between IoT devices without interferences are extremely critical. Still, securing data exchanges is necessary to avoid losing or compromising privacy. The increased number of smart things around us with sensitive data necessitates a transparent and easy access control management in such a way that for example one vendor can just read the data while another is allowed to control the device. In this regard, some solutions have been proposed such as grouping embedded devices into virtual networks and only present desired devices within each virtual network. Another approach is to support access control in the application layer on a per- vendor basis.

Management:

The connection of billions or trillions of smart devices presents service providers with daunting issues to manage the Fault, Configuration, Accounting, Performance and security (FCAPS) aspects of these devices. This management effort necessitates the development of new light-weight management protocols to handle the potential management nightmare that will potentially stem from the deployment of the IoT in the coming years. Managing IoT devices and applications can be an effective factor for growing the IoT deployments. For example, monitoring the M2M communication of the IoT objects is important to ensure all times connectivity for providing on demand services.

The Light-weight M2M (LWM2M) is a standard that is being developed by the Open Mobile Alliance to provide interface between M2M devices and M2M Servers to build an application agnostic scheme for the management of a variety of devices. It aims to provide M2M applications with remote management capabilities of machine-to-machine devices, services, and applications.

The NETCONF Light protocol is an Internet Engineering Task Force (IETF) effort for the management of constrained devices provides mechanisms to install, manipulate, and delete the configuration of network devices. It is capable of managing a broad range of devices from resource- constrained to resource-rich devices.

The independently developed MASH IoT Platform is an example of a platform that facilitates the management (monitoring, control, and configuration) of IoT assets anywhere in real-time using an IoT dashboard on smart phones. Maintaining compatibility across the IoT layers also needs to be managed to enhance connectivity speed and to ensure service delivery. The Open Mobile Alliance (OMA) Device Management working group is specifying protocols and mechanisms for the management of mobile devices and services in resource constrained environments.

Manufacturing:

Manufacturing challenges must be convincingly solved. Costs must be lowered to less than one cent per passive RFID tag, and production must reach extremely high volumes, while the whole production process must have a very limited impact on the environment, be based on strategies for reuse and recycling considering the overall life-cycle of digital devices and other products that might be tagged or sensor-enabled.

Barriers:

But there are also existing barriers for the IoT, especially in the field of regulations, security and safety. Main goal is to better protect the privacy of people and force companies to establish secure ways to manage data and information.

Absence of Governance:

One major barrier for the widespread adoption of the Internet of Things technology is the absence of governance. Without an impartial governing authority it will be impossible to have a truly global IoT, accepted by states, companies, trade organizations and the common people.

Today there is not a unique universal numbering scheme: EPC global and the Ubiquitous Networking Lab propose two different, non-compatible ways of identifying objects and there is the risk to have them competing in the coming future over the global market. There is also the need of keeping governance as generic as possible, as having one authority per application field will certainly lead to overlap, confusion and competition between standards. Objects can have different identities in different contexts so having multiple authorities would create a kind of multi-homing, which can lead to disastrous results.

Privacy and Security:

In order to have a widespread adoption of any object identification system, there is a need to have a technically sound solution to guarantee privacy and the security of the customers. While in many cases the security has been done as an add-on feature, it is the feeling that the public acceptance for the Internet of Things will happen only when the strong security and privacy solutions are in place. In particular, attacks have to be intercepted, data authenticated, access controlled and the privacy of customers (natural and legal persons) guaranteed. This could be hybrid security mechanisms that for example combine hardware security with key diversification to deliver superior security that makes attacks significantly more difficult or even impossible.

The selection of security features and mechanisms will continue to be determined by the impact on business processes; and trade-offs will be made between chip size, cost, functionality, interoperability, security, and privacy.

The security and privacy issues should be addressed by the forthcoming standards which must define different security features to provide confidentiality, integrity, or availability services. There are also a range of issues related to the identity of people. These must be dealt with in politics and legislation, and they are of crucial importance for the efficient public administrations of the future.

IoT Reference Architecture

5.1. IoT Architecture Overview:

IoT can be classified into a four or five-layered architecture which gives you a complete overview of how it works in real life. The various components of the architecture include the following:

Four-layered architecture: this includes media/device layer, network layer, service and application support layer, and application layer.

Five-layered architecture: this includes perception layer, network layer, middleware layer, application layer, and business layer.

5.2. Functions of Each Layer:

Sensor/Perception layer: This layer comprises of wireless devices, sensors, and radio frequency identification (RFID) tags that are used for collecting and transmitting raw data such as the temperature, moisture, etc. which is passed on to the next layer.

Network layer: This layer is largely responsible for routing data to the next layer in the hierarchy with the help of network protocols. It uses wired and wireless technologies for data transmission.

Middleware layer: This layer comprises of databases that store the information passed on by the lower layers where it performs information processing and uses the results to make further decisions.

Service and application support layer: This layer involves business process modeling and execution as well as IoT service monitoring and resolution.

Application layer: It consists of application user interface and deals with various applications such as home automation, electronic health monitoring, etc.

Business layer: this layer determines the future or further actions required based on the data provided by the lower layers.

5.3. Building an IoT Architecture:

Four things form basic building blocks of the IoT system – sensors, processors, gateways, applications. Each of these nodes has to have its own characteristics in order to form a useful IoT system.

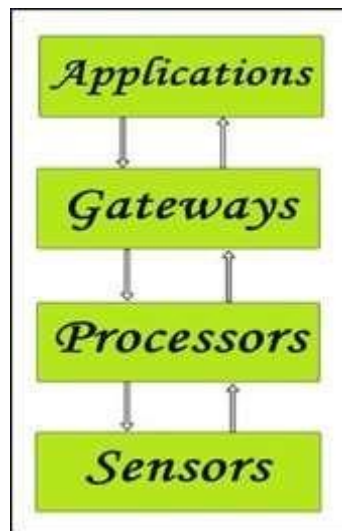


Fig: Basic building blocks of the IoT Sensors

These form the front end of the IoT devices. These are the so-called “Things” of the system. Their main purpose is to collect data from its surroundings (sensors) or give out data to its surrounding (actuators). These have to be uniquely identifiable devices with a unique IP address so that they can be easily identifiable over a large network. These have to be active in nature which means that they should be able to collect real-time data. These can either work on their own (autonomous in nature) or can be made to work by the user depending on their needs (user-controlled).

Examples of sensors are gas sensor, water quality sensor, moisture sensor, etc.

Processors:

Processors are the brain of the IoT system. Their main function is to process the data captured by the sensors and process them so as to extract the valuable data from the enormous amount of raw data collected. In a word, we can say that it gives intelligence to the data. Processors mostly work on real-time basis and can be easily controlled by applications. These are also responsible for securing the data – that is performing encryption and decryption of data. Embedded hardware devices, microcontroller, etc are the ones that process the data because they have processors attached to it.

Gateways:

Gateways are responsible for routing the processed data and send it to proper locations for its (data) proper utilization. In other words, we can say that gateway helps in to and fro communication of the data. It provides network connectivity to the data. Network connectivity is essential for any IoT system to communicate. LAN, WAN, PAN, etc are examples of network gateways.

Applications:

Applications form another end of an IoT system. Applications are essential for proper utilization of all the data collected. These cloud-based applications which are responsible for rendering the effective meaning to the data collected. Applications are controlled by users and are a delivery point of particular services. Examples of applications are home automation apps, security systems, industrial control hub, etc.

5.4. Main design principles of IoT:

1. Do your research

When designing IoT-enabled products, designers might make the mistake of forgetting why customers value these products in the first place. That's why it's a good idea to think about the value an IoT offering should deliver at the initial phase of your design. When getting into IoT design, you're not building products anymore. You're building services and experiences that improve people's lives. That's why in-depth qualitative research is the key to figuring out how you can do that. Assume the perspective of your customers to understand what they need and how your IoT implementation can solve their pain points. Research your target audience deeply to see what their existing experiences are and what they wish was different about them.

2. Concentrate on value

Early adopters are eager to try out new technologies. But the rest of your customer base might be reluctant to put a new solution to use. They may not feel confident with it and are likely to be cautious about using it. If you want your IoT solution to become widely adopted, you need to focus on the actual tangible value it's going to deliver to your target audience. What is the real end-user value of your solution? What might be the barriers to adopting new technology? How can your solution address them specifically? Note that features the early tech adopters might find valuable might turn out to be completely uninteresting for the majority of users. That's why you need to carefully plan which features to include and in what order, always concentrating on the actual value they provide.

3. Don't forget about the bigger picture

One characteristic trait of IoT solutions is that they typically include multiple devices that come with different capabilities and consist of both digital and physical touch points. Your solution might also be delivered to users in cooperation with service providers. That's why it's not enough to design a single touch point well. Instead, you need to take the bigger

picture into account and treat your IoT system holistically. Delineate the role of every device and service. Develop a conceptual model of how users will perceive and understand the system. All the parts of your system need to work seamlessly together. Only then you'll be able to create a meaningful experience for your end-users.

4. Remember about the security

Don't forget that IoT solutions aren't purely digital. They're located in the real-world context, and the consequences of their actions might be serious if something goes wrong. At the same time, building trust in IoT solutions should be one of your main design drivers. Make sure that every interaction with your product builds consumer trust rather than breaking it. In practice, it means that you should understand all the possible error situations that may be related to the context of its use. Then try to design your product in away to prevent them. If error situations occur, make sure that the user is informed appropriately and provided with help. Also, consider data security and privacy as a key aspect of your implementation. Users need to feel that their data is safe, and objects located in their workspaces or home can't be hacked. That's why quality assurance and testing the system in the real-world context are so important.

5. Build with the context in mind

And speaking of context, it pays to remember that IoT solutions are located at the intersection of the physical and digital world. The commands you give through digital interfaces produce real-world effects. Unlike digital commands, these actions may not be easily undone. In a real-world context, many unexpected things may happen. That's why you need to make sure that the design of your solution enables users to feel safe and in control at all times. The context itself is a crucial consideration during IoT design. Depending on the physical context of your solution, you might have different goals in mind. For example, you might want to minimize user distraction or design devices that will be resistant to the changing weather conditions. The social context is an important factor, as well. Don't forget that the devices you design for workspaces or homes will be used by multiple users.

6. Make good use of prototypes

IoT solutions are often difficult to upgrade. Once the user places the connected object somewhere, it might be hard to replace it with a new version – especially if the user would have to pay for the upgrade. Even the software within the object might be hard to update because of security and privacy reasons. Make sure that your design practices help to avoid costly hardware iterations. Get your solution right from the start. From the design

perspective, it means that prototyping and rapid iteration will become critical in the early stages of the project.

5.5. Standards consideration for IoT:

Alliances have been formed by many domestic and multinational companies to agree on common standards and technology for the IoT. However, no universal body has been formed yet. While organizations such as IEEE, Internet Engineering Task Force (IETF), ITU-T, OneM2M, 3GPP, etc., are active at international level, Telecommunication Standards Development Society, India (TSDSI), Global ICT Standardization Forum for India (GISFI), Bureau of Indian Standards (BIS), Korean Agency for Technology and Standards (KATS), and so on, are active at national level and European Telecommunications Standards Institute (ETSI) in the regional level for standardization.

5.6. IoT Architecture:

The Internet of Things (IoT) has seen an increasing interest in adaptive frameworks and architectural designs to promote the correlation between IoT devices and IoT systems. This is because IoT systems are designed to be categorized across diverse application domains and geographical locations. It, therefore, creates extensive dependencies across domains, platforms and services. Considering this interdependency between IoT devices and IoT systems, an intelligent, connection-aware framework has become a necessity; this is where IoT architecture comes into play! Imagine a variety of smart IoT systems from sensors and actuators to internet gateways and Data Acquisition Systems all under the centralized control of one “brain”! The brain here can be referred to as the IoT architecture, whose effectiveness and applicability directly correlate with the quality of its building blocks.

The way a system interacts and the different functions an IoT device performs are various approaches to IoT architecture. Since we can call the architecture the brain, it’s also possible to say that the key causes of poor integration in IoT systems are the shortage of intelligent, connection-aware architecture to support interaction in IoT systems.

An IoT architecture is the system of numerous elements that range from sensors, protocols, actuators, to cloud services, and layers. Besides, devices and sensors the Internet of Things (IoT) architecture layers are distinguished to track the consistency of a system through protocols and gateways. Different architectures have been proposed by researchers and we can all agree that there is no single consensus on architecture for IoT. The most basic architecture is three-layer architecture.

State-of-the-art:

The IoT can be considered both a dynamic and global networked infrastructure that manages self- configuring objects in a highly intelligent way. This, in turn, allows the interconnection of IoT devices that share their information to create new applications and services which can improve human lives. Originally, the concept of the IoT was first introduced by Kevin Ashton, who is the founder of MIT auto-identification centre in 1999. Ashton has said, “The Internet of Things has the potential to change the world, just as the Internet did. May be even more so”. Later, the IoT was officially presented by the International Telecommunication Union (ITU) in 2005.

The IoT has many definitions suggested by many organizations and researchers. However, the definition provided by ITU in 2012 is the most common. It stated: “a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on, existing and evolving, interoperable information and communication technologies”.

In addition, Guillemin and Friess in have suggested one of the simplest definitions that describe the IoT in a smooth manner. It stated: “The Internet of Things allows people and things to be connected Anytime, Any place, with anything and anyone, ideally using any path/network and any service”. Several definitions were suggested by many researchers describing the IoT system from different perspectives but the important thing that majority or researchers have agreed on is the IoT is created for a better world for all the human beings.

The IoT is a promising technology that starts to grow significantly. There were already more objects/things connected to the Internet than people from 2008. Predictions are made that in the future; the number of Internet-connected devices will reach or even exceed 50 billion. In addition, the IoT becomes the most massive device market that enables companies to save billions of dollars. It has added \$1.7 trillion in value to the global economy in 2019. This involves hardware, software, management services, installation costs, and economic value from realized IoT efficiencies.

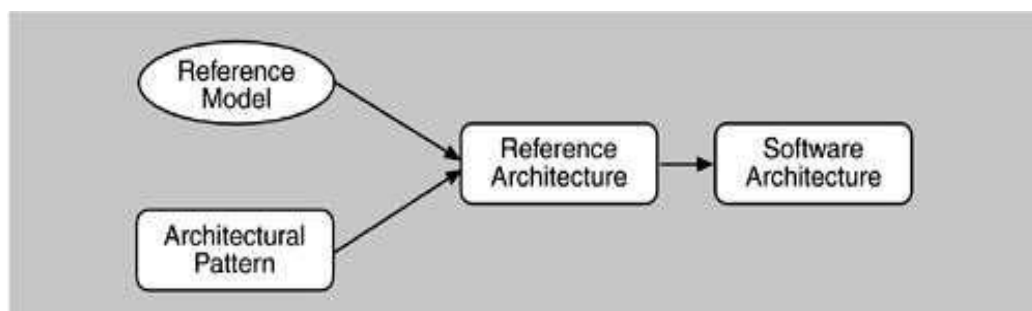
Nowadays, the IoT notion has evolved to include the perception of realizing a global infrastructure of interconnected networks of physical and virtual objects. The huge technological development has expanded the idea of the IoT to involve other technologies such as Cloud computing and Wireless Sensor Networks (WSNs). The

IoT has become able to connect both humans and things anywhere, and anytime, ideally using any path/network. The IoT has become one of the interesting topics to many researchers. According to Google, the number of IoT journal and conference papers has almost doubled from 2004 to 2010. From 2010, the IoT articles are dramatically increased to reach about 985 articles in 2015.

5.7. Architecture Reference Model:

A reference model is a division of functionality together with data flow between the pieces. A reference model is a standard decomposition of a known problem into parts that cooperatively solve the problem. Arising from experience, reference models are a characteristic of mature domains. Can you name the standard parts of a compiler or a database management system? Can you explain in broad terms how the parts work together to accomplish their collective purpose? If so, it is because you have been taught reference model of these applications.

A reference architecture is a reference model mapped onto software elements (that cooperatively implement the functionality defined in the reference model) and the data flows between them. Whereas a reference model divides the functionality, a reference architecture is the mapping of that functionality onto a system decomposition. The mapping may be, but by no means necessarily is, one to one. A software element may implement part of a function or several functions. Reference models, architectural patterns, and reference architectures are not architectures; they are useful concepts that capture elements of an architecture. Each is the outcome of early design decisions. The relationship among these design elements is shown in Figure.



5.8. IoT Reference Architecture:

The reference architecture consists of a set of components. Layers can be realized by means of specific technologies, and we will discuss options for realizing each component.

There are also some cross- cutting/vertical layers such as access/identity management.

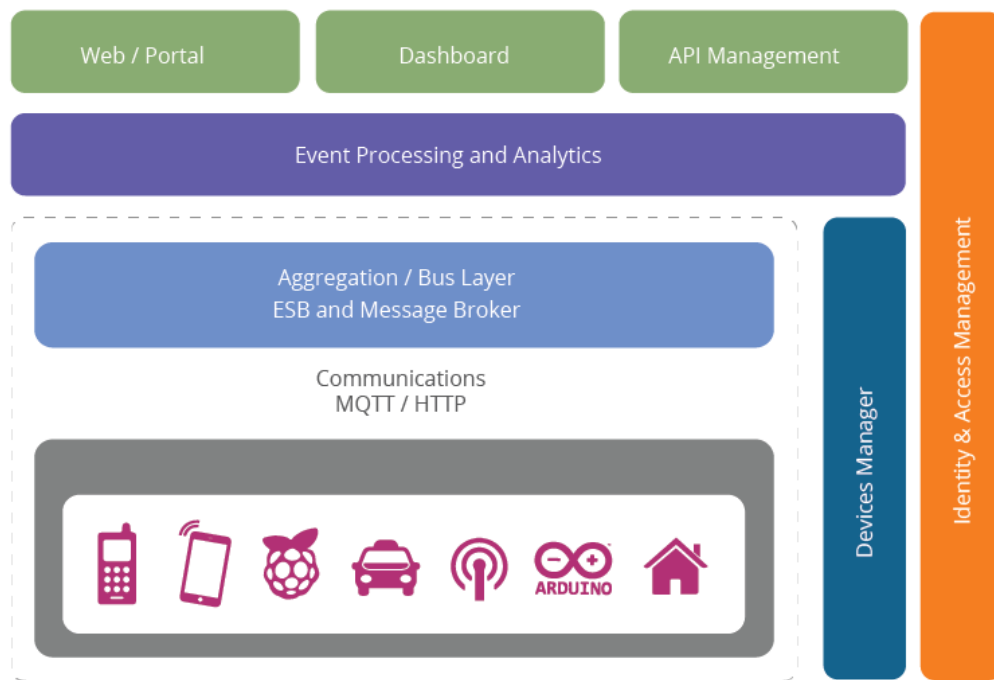


Fig: IoT Reference Architecture

5.9. Industrial Automation:

Industrial automation is all about intelligent process control. The IoT doesn't depend on your hardware because you can choose independent control systems, sensors, and network components. The IoT's power goes beyond the limited features and functionalities your device manufacturer or software provider offers. Those who use the IoT in industrial automation processes can connect multiple sites and locations so that they operate in harmony. Industrial automation is well-known for diverting technology from the commercial sphere and adapting it to new ends. The industry's widespread IoT adoption builds upon this tested concept in numerous ways:

Wireless Improvement of Existing Monitoring and Control Systems:

Wireless connectivity makes it simpler to implement complex control systems in awkward, remote, or hazardous environments. For instance, using wired networks to link remotely controlled cranes, robot arms, and other manufacturing devices can be problematic due to their unique ranges of motion and exposure to harsh fabrication environments. The IoT's compatibility with wireless technology lets enterprises replace standard linkages with fully enclosed mesh radios that perform the same functions. Even better, these alternatives may be more useful for automation processes that require fine-tuning or ongoing

adjustments. For instance, you don't have to replace miles of Ethernet cable to achieve higher transmission speeds with Wi-Fi.

Building Factories That Build and Run Themselves:

Growth has decided on the pros and cons. Although few experiences beat the thrill of taking your organizational training wheels off and cruising along, doing business at a higher volume introduces unique risks, such as the potential for greater waste should you take a wrong turn. A company that wanted to conserve resources might use an industrial sensor system to tell it when to shut down auxiliary production lines. An enterprise that relies on automated stock machines to transport replacement parts to workstations could employ a connected framework to initiate new deliveries without waiting for approval from a line manager. The IoT also makes it possible to create digital twins. These replicas of existing systems serve as test beds for new projects and experiments.

Managing Communications Whenever the Need Arises:

The IoT enhances traditional automation schemes by making everything on-demand. When you make a change from a control dashboard, you get to see its effects ripple outward right away. What you might not expect is that the system also performs the innumerable tedious tasks that facilitate good digital communication, such as:

- ❖ Rerouting traffic to keep data moving no matter how much information happens to be passing through,
- ❖ Accounting for the effects of network topologies to sustain optimized service quality,
- ❖ Accommodating vendor-neutral communication protocols and schemes to support a wider variety of hardware and software,
- ❖ Self-detecting equipment failures and automatically switching to functional network elements, and
- ❖ Duplicating and storing data as necessary to prevent catastrophic losses.

Although this kind of work may get overlooked because it goes on in the background, it's an essential part of ensuring that automation frameworks behave deterministically. When your industrial communication systems behave consistently, sound management practices prove easier to execute.

Decentralizing Debugging and Maintenance:

There's no shortage of industrial automation maintenance philosophies to choose from, so debugging can get confusing. IoT mesh networks help stakeholders handle maintenance more logically. You can debug, tweak, and maintain controllers and sensors from local network nodes to cut down on overhead and make the best use of limited bandwidth. Decentralized maintenance is the glue that helps automation systems stick together and run seamlessly even as they expand. By using the IoT to program functions at the node level, you can optimize resource usage and slash costs for a more productive enterprise.

Investing in the IoT in Industrial Automation Settings:

Internet of Things technologies offer a spectrum of other potential benefits that we haven't even covered. There are voice-recognition systems that let factory owners authenticate themselves and implement complex behaviors without any manual programming. Embedded and linked networks contribute to improved lifecycle oversight, demand-specific customization, and better cost control, but choosing the best-equipped IoT layout and technical components can be a tough task. Optimality isn't universal. It's defined by the circumstances, so you need to move forward with an eye on building something that's sufficiently flexible yet robust enough to survive the unexpected.

Interaction and Remote Control:

IoT devices produce many types of information, including telemetry, metadata, state, and commands and responses. Telemetry data from devices can be used in short operational timeframes or for longer-term analytics and model building. (For more on this diversity, read the overview of Internet of Things.)

Many devices support local monitoring in the form of a buzzer or an alarm panel on-premises. This type of monitoring is valuable but has limited scope for in-depth or long-term analysis. This article instead discusses remote monitoring, which involves gathering and analyzing monitoring information from a remote location using cloud resources. Operational and device performance data is often in the form of a time series, where each piece of information includes a time stamp. This data can be further enriched with dimensional labels (sometimes referred to as tags), such as labels that identify hardware revision, operating time zone, installation location, firmware version, and so on.

Time-series telemetry can be collected and used for monitoring. Monitoring in this context refers to using a suite of tools and processes that help detect, debug, and resolve problems that occur in systems while those systems are operating. Monitoring can also give you insight into the systems and help improve them.

The state of monitoring IT systems, including servers and services, has continuously improved. Monitoring tools and practices in the cloud-native world of micro services and Kubernetes are excellent at monitoring based on time-series metric data. These tools aren't designed specifically for monitoring IoT devices or physical processes, but the constituent parts—labeled series of metrics, visualization, and alerts—all can apply to IoT monitoring.

Remoteness:

Unlike servers in a cluster, monitored devices might be far from the systems that are organizing the metric data and providing visualizations. There is debate in the monitoring community about push-based versus pull-based collection methods for monitoring telemetry. For IoT devices, push-based monitoring can be more convenient. But you must consider the trade-offs in the entire stack (including things like the power of the query language, and the efficiency and cost of the time-series storage) when you choose which metrics framework to use.

In either approach, a remote device might become disconnected from the monitoring system. No effective monitoring can occur if data isn't flowing. Stale and missing metrics can hamper the value of a metric series where you might be calculating rates or other types of values derived over time. When you're monitoring remote devices, it's also important to recognize that variation in timestamps is possible and to ensure the best clock synchronization possible. The following diagram shows a schematic of remote devices, with centralized monitoring compared to cluster-based monitoring.

Monitoring design patterns:

When it is determined which systems you're monitoring, you need to think about why you're monitoring. The system you're working with is providing a useful function, and the goal of monitoring is to help ensure that a function or service is performing as intended.

When monitoring software services, you look for measurements around the performance of that service, such as web request response times. When the service is a physical process such as space heating, electrical generation, or water filtration, you might use devices to instrument that physical process and take measurements of things like engine hours or cycle times. Whether you're using a device as a means solely to instrument a

physical process, or whether the device itself is performing a service, you want to have a number of measurements about the device itself.

Measurements made at the point of instrumentation result in a metric being sent and recorded in the centralized monitoring system. Metrics might be low level (direct and unprocessed) or high level (abstract). Higher-level metrics might be computed from lower-level metrics. One should start by thinking about the high-level metrics you need in order to ensure delivery of service. One can then determine which lower-level metrics you need to collect in order to support your monitoring goals. Not all metrics are useful, and it's important not to fall into the trap of measuring things just because you can, or because they look impressive (so-called "vanity metrics").

Good metrics have the following characteristics:

- ❖ They're actionable. They inform those who operate or revise the service when they need to change its behavior.
- ❖ They're comparative. They compare the performance of something over time, or between groups of devices whose members are in different location or have different firmware or hardware versions.
- ❖ They're understandable and relevant in an operational context. This means that in addition to raw values like totals, they can provide information like ratios and rates.
- ❖ They provide information at the right resolution. You can choose how often you sample, how often you report, and how you average, bin, and graph your metrics. These values all need to be chosen in the domain context of the service you're trying to deliver. For example, providing 1-second reporting on an IoT device's SD card capacity generates a lot of unnecessary detail and volume. And looking only at CPU load averaged per hour will absorb and hide short, service-crushing spikes in activity. There might be periods of troubleshooting where you dial up the fidelity of metrics for better diagnostics. But the baseline resolution should be appropriate for what you need in order to meet your monitoring needs.
- ❖ They illuminate the difference between symptoms, causes, and correlations across what you're measuring. Some measurements are leading indicators of a problem, and you might want to build alerting on those. Other measurements are lagging indicators and help you understand what has happened; these measurements

are often used for exploratory analysis.

Chapter – 06

Future of IoT

It is possible to identify, for the years to come, four distinct macro-trends that will shape the future of internet technologies, together with the explosion of ubiquitous devices that constitute the future Internet of Things:

1. The first one, sometimes referred as “exaflood” or “data deluge”, is the explosion of the amount of data collected and exchanged. As current networks are ill-suited for this exponential traffic growth, there is a need by all the actors to re-think current networking and storage architectures. It will be imperative to find novel ways and mechanisms to find, fetch, and transmit data. One relevant reason for this data deluge is the explosion in the number of devices collecting and exchanging information as envisioned as the Internet of Things becomes a reality.

The term **exaflood**, coined by Bret Swanson of Progress & Freedom Foundation, refers to the growing torrent of data on the Internet.

2. The energy required to operate the intelligent devices will dramatically decrease. Already today many data centers have reached the maximum level of energy consumption and the acquisition of new devices has necessarily to follow the dismissal of old ones. Therefore, the second trend can be identified covering all devices and systems from the tiniest smart dust to the huge data centers: the search for a zero level of entropy where the device or system will have to harvest its own energy.

3. Miniaturization of devices is also taking place amazingly fast. The objective of a single-electron transistor is getting closer, which seems the ultimate limit, at least until new discoveries in physics.

4. Another important trend is towards autonomic resources. The ever growing complexity of systems will be unmanageable, and will hamper the creation of new services and applications, unless the systems will show self-* properties, such as self-management, self-healing and self-configuration.

As a general trend, as it becomes less expensive to integrate technology into physical objects, we will see more application and adoption of IoT. In consequence, IoT will have major implications for both business-to-business and business-to-consumer companies in the next years.

6.1. IoT Trends and Predictions:

Before we dive into the details of trends in IoT, let's take a sneak peek at some top trends and predictions that you can expect as influencers in IoT technology.

1. Digital Twins

With widespread digitization and the availability of sensor technology, 2024 is expected to see a rapid proliferation of digital twins in more industries. The technology will offer more visibility into organizations' processes and help stakeholders analyze and improve them. The last two years have seen the rapid maturation of this technology, and the trend is expected to become more mainstream. Organizations are trying to build robust supply chains and logistics solutions after the COVID-19 crisis, and digital twins will play a huge role here.

Companies in the space are collaborating with tech giants to reach diverse clients and improve their solutions. For instance, Ansys Inc., involved in mechanical engineering and simulation, has joined hands with Microsoft Azure.

The past year has also seen attempts at building a standard for digital twins, with Twinify and Digital Twin Consortium leading the pack.

2. AI and IoT



The proliferation of IoT devices has allowed organizations previously unavailable access to granular data about their operations. Businesses can get detailed insights regarding their employees, productivity, service or product quality, and other information. Companies have started to leverage this data to develop and power their artificial intelligence models.

By integrating AI into their systems, businesses can automate more processes and improve the quality of their products and services. The last year has seen more IoT solutions with edge processing capabilities.

These solutions allow organizations to deploy AI closer to the sensors or actuators, reducing connectivity requirements and empowering faster decision-making within Internet of Things systems. They have also enabled the deployment of IoT solutions in more industries and for more applications.

We have already seen technologies like predictive maintenance become widespread and mainstream. In 2024, we can expect more businesses to adopt these technologies in their manufacturing, assembly, logistics, shipping, fleet management, and other processes.

3. New Sensor Tech

According to Gartner, we can expect innovations in sensor tech throughout 2024. We can also expect price drops in existing sensor tech and a subsequent reduction in the price of IoT solutions. Gartner also predicts innovation in algorithms which will help gain more insights from existing sensor technology.

At CES 2024, Bosch is expected to present sensors such as long-range Lidars, high-performance magnetometers, barometric pressure sensors, and AI-based sensor systems.

The company, along with many other players, is expected to showcase new sensor systems designed for autonomous vehicles. There is news about sensors that can track driver and vehicle performance, preventive maintenance, and fleet analytics.

In the field of smart agriculture, Himax Technologies has partnered with Sseed Studio to showcase the LoRaWAN Vision AI sensor at CES.

3. 5G Connected Car Services



In the coming year, we can expect 5G connectivity to become more common globally. The last year saw many providers launch their 5G services, but they were only available in a few locations.

Many providers were also criticized for poor implementation of the 5G standard. However, in 2024, we are expecting service providers to fix these issues and for the technology to reach more people.

The launch of 5G services has also prompted advances in connected car tech. Many operators are trying to launch connected car services as an add-on to customers' 5G plans. This is expected to increase vehicle-to-vehicle communication, improve ADAS systems' situational awareness, and reduce collisions.

AT&T has already announced that it will work with BMW and Polaris to offer connected vehicle tech. Their Number Sync plan allows owners of select BMW cars to make and receive calls through their vehicles even when they don't have their phones with them.

5. Chip-Level Innovation for IoT Devices

While we have seen advances in IoT sensors and solutions, we have yet to see much innovation at the chip level. Over the next years, we can expect this to change. We can expect more specialized chips for IoT systems leveraging their specific advantages and use cases.

These changes are primarily driven by the popularity of edge computing and AI in IoT solutions. Organizations are demanding more processing power at the edge, and the industry is responding with new architectures and instruction sets.

we can also expect specialized chips that can handle neural networks and AI systems at low power. And, of course, these changes will drive further innovation in AI solutions that will take advantage of the new chips.

Organizations and governments are also exploring solutions that can reduce the global chip shortage, which is expected to continue further. Companies are trying to build robust supply chains and reduce their reliance on a very small number of chip manufacturers.

6. IoT in Healthcare

The last couple of years have seen IoT advance into healthcare in the form of wearables, voice assistants, and connected medical equipment. Heart rate and SpO2 sensors have become common in smartwatches and wearables. In 2024, we can expect more IoT solutions to hit healthcare institutions. The market is expected to hit \$267 billion by the following year.

The upcoming year will see wearables get smarter and incorporate sensors that can measure more parameters. We can also expect more at-home smart sensors to monitor patients' vitals.

According to Forbes, we may become familiar with the concept of a virtual hospital ward, where doctors and nurses will oversee and monitor patients within their homes.

7. Ethical, Legal, and Privacy Concerns About IoT



The past couple of years has seen increasing awareness about the risks of IoT devices among the general population. This was highlighted in Gartner's 'Top IoT Strategic Trends and Technologies Through 2024' report. As the technology matures, the ethical and social implications have become more apparent.

Consumers have become aware of the privacy risks that IoT devices pose. Concerns about the ownership of data, how it may be used, and how it will be secured have been raised. For instance, over the last year, it became clear that Ring doorbells were sharing footage with law enforcement agencies without users' consent, raising questions about privacy and consent.

Governments are introducing legislation to address these concerns. The EU's Digital Services Act is one such measure aimed at reducing the impact of malicious actors. In the US, the Federal Trade Commission has proposed new regulations for data privacy, and California has introduced new privacy laws to enhance consumer rights.

By the end of 2024, we can expect more privacy regulations that impact how IoT solutions are designed and used.

8. IoT for Sustainability

Climate change has been on the agenda for governments and organizations around the world. Over the last couple of years, organizations have started using IoT solutions to reduce their environmental footprint. Solutions such as predictive maintenance, energy-efficient IoT devices, and connected sensors are expected to become more popular.

From 2024, more organizations will look at IoT solutions to increase their energy efficiency. We may also see widespread adoption of solutions such as smart waste management, precision agriculture, and renewable energy management systems.

Organizations are likely to invest in smart buildings and smart grids to support sustainability goals. Smart buildings will help monitor and manage energy consumption and optimize the use of resources. Smart grids will enable real-time management of electricity distribution and improve the efficiency of renewable energy sources.

9. 6G Research

While 5G is still in its early stages, research on the next generation of connectivity—6G—is well underway. From 2024, we can expect more breakthroughs in 6G technology and discussions about its potential impact on IoT.

The 6G technology promises higher speeds, lower latency, and more reliable connections. These advancements are expected to enhance the capabilities of IoT systems and drive the development of new applications and use cases.

Several organizations and research institutions are already working on 6G technologies. For instance, Samsung has released a white paper on 6G, outlining its vision for the technology and its potential applications.

As research progresses, we may see the first prototypes and experimental deployments of 6G technology in the coming years.

10. Quantum Computing

The rise of quantum computing is expected to have a significant impact on IoT in the future. While quantum computing is still in its early stages, it has the potential to revolutionize how data is processed and analyzed.

Quantum computers can solve complex problems much faster than traditional computers. This capability could be applied to various IoT use cases, such as optimizing supply chains, enhancing security, and analyzing large datasets.

From 2024, we can also expect more advancements in quantum computing and its integration with IoT systems. Organizations may start exploring quantum computing solutions to address specific challenges and unlock new opportunities.

6.2. The Role of AI and Machine Learning in IoT:

The Internet of Things (IoT) has made smart homes & cities possible, but the true potential of IoT lies in processing the massive amounts of data generated by connected devices. Here's where AI & ML come into play. AI & ML are computer systems that can learn from data & make decisions without explicit instructions. In IoT, they can automate processes, optimize efficiencies & provide predictive insights.

AI & ML have numerous use cases in IoT, such as streamlining supply chains, optimizing building management, or predicting machine maintenance. They can also improve the security of IoT devices & networks. AI & ML algorithms can detect threats & vulnerabilities, prevent attacks, & identify anomalous behavior. They can also contribute to sustainability efforts, such as reducing energy waste & lowering carbon footprints.

AI & ML are critical components in the future of IoT. They hold the key to creating smarter & more efficient systems, improving security & creating a more sustainable future. As IoT data continues to grow, AI & ML will play an increasingly significant role, making it imperative that professionals upskill with Knowledge Hut's Software Development training to stay at the forefront and increase your value in the job market.







Arduino for Beginners

Arduino is a project, open-source hardware, and software platform used to design and build electronic devices. It designs and manufactures microcontroller kits and single-board interfaces for building electronics projects.

The Arduino boards were initially created to help the students with the non-technical background.

The designs of Arduino boards use a variety of controllers and microprocessors.

The Arduino board consists of sets of analog and digital I/O (Input / Output) pins, which are further interfaced to breadboard, expansion boards, and other circuits. Such boards feature the model, Universal Serial Bus (USB), and serial communication interfaces, which are used for loading programs from the computers.

It also provides an IDE (Integrated Development Environment) project, which is based on the Processing Language to upload the code to the physical board.

The projects are authorized under the GPL and LGPL. The GPL is named as GNU General Public License. The licensed LGPL is named as GNU Lesser General Public License. It allows the use of Arduino boards, its software distribution, and can be manufactured by anyone.

It is also available in the form of self practicing kits.

The Arduino is used for various purposes, such as:

- ❖ Finger button
- ❖ Button for motor activation
- ❖ Light as a sensors
- ❖ LED button
- ❖ Designing
- ❖ The Building of electronic devices

What is Arduino?

Arduino is a software as well as hardware platform that helps in making electronic projects. It is an open source platform and has a variety of controllers and microprocessors. There are various types of Arduino boards used for various purposes.

The Arduino is a single circuit board, which consists of different interfaces or parts. The board consists of the set of digital and analog pins that are used to connect various devices and components, which we want to use for the functioning of the electronic devices.

Most of the Arduino consists of 14 digital I/O pins.

The analog pins in Arduino are mostly useful for fine-grained control. The pins in the Arduino board are arranged in a specific pattern. The other devices on the Arduino board are USB port, small components (voltage regulator or oscillator), microcontroller, power connector, etc.

7.1. Features:

The features of Arduino are listed below:

- ❖ Arduino programming is a simplified version of C++, which makes the learning process easy.
- ❖ The Arduino IDE is used to control the functions of boards. It further sends the set of specifications to the microcontroller.
- ❖ Arduino does not need an extra board or piece to load new code.
- ❖ Arduino can read analog and digital input signals.
- ❖ The hardware and software platform is easy to use and implement.

7.2. History:

The project began in the Interaction Design Institute in Ivrea, Italy. Under the supervision of Casey Reas and Massimo Banzi, the Hernando Bar in 2003 created the Wiring (a development platform). It was considered as the master thesis project at IDII. The Wiring platform includes the PCB (Printed Circuit Board). The PCB is operated with the ATmega168 Microcontroller.

The ATmega168 Microcontroller was an IDE. It was based on the library and processing functions, which are used to easily program the microcontroller.

In 2005, Massimo Banzi, David Cuartielles, David Mellis, and another IDII student supported the ATmega168 to the Wiring platform. They further named the project as Arduino.

The project of Arduino was started in 2005 for students in Ivrea, Italy. It aimed to provide an easy and low-cost method for hobbyists and professionals to interact with the environment using the actuators and the sensors. The beginner devices were simple motion detectors, robots, and thermostats.

In mid-2011, the estimated production of Arduino commercially was 300,000. In 2013, the Arduino boards in use were about 700,000.

Around April 2017, Massimo Banzi introduced the foundation of Arduino as the "new beginning for Arduino". In July 2017, Musto continued to pull many Open Source licenses and the code from the websites of the Arduino. In October 2017, Arduino introduced its collaboration with the ARM Holdings. The Arduino continues to work with architectures and technology vendors.

Microcontroller:

The most essential part of the Arduino is the Microcontroller, which is shown below:



- ❖ Microcontroller is small and low power computer. Most of the microcontrollers have a RAM (Random Access Memory), CPU (Central Processing Unit), and a memory storage like other computer systems.
- ❖ It has very small memory of 2KB (two Kilobytes). Due to less memory, some microcontrollers are capable of running only one program at a time.

- ❖ It is a single chip that includes memory, Input/Output (I/O) peripherals, and a processor.
- ❖ The GPIO (General Purpose Input Output) pins present on the chip help us to control other electronics or circuitry from the program.

Electronic devices around us

We have many electronic devices around us. Most of the appliance consists of the microcontroller for its functioning. Let's discuss some of the examples.

- ❖ Microcontroller present in Microwave Oven accepts the user input and controls the magnetron that generates microwave rays to cook the food and displays the output timer.
- ❖ Modern cars also contain dozens of microcontrollers working in tandem (one after another) to control functions like lighting, radio interface, etc.

Projects

Let's consider a simple project of LED blink.

We need a software to install our sketch or code to the Arduino board. The LED will blink after the successful uploading of code. The software is called as Arduino IDE (Integrated Development Environment).

There are various projects created with the help of the Arduino. Some of the projects are listed below:

- ❖ Home Automation System using IOT (Internet of Things).
- ❖ Solar Power water trash collector.
- ❖ Fire Detector, etc.

Some projects require a list of components. So, for easy convenience and hands-on projects, the Arduino kits are available easily in market.

Arduino Kits

We can easily start with our electronics projects using the complete kit. It also helps us to create hands-on and engaging projects.

Some of the popular Arduino kits are listed below:

- ✓ Arduino Starter kit
- ✓ Robot Linking UNO kit for learning
- ✓ Arduino UNO 3 Ultimate starter kit
- ✓ UNO Super starter kit
- ✓ Mega 2560 Starter Kit

Arduino IDE:

The IDE makes the traditional projects even easier and simpler. The USB cable is used to load the program or sketch on the specific Arduino board.



The IDE application is suitable for Windows, Mac OS X, and Linux. It supports the programming language C and C++. We need to connect the Genuino and Arduino board with the IDE to upload the sketch written in the Arduino IDE software.

Many other companies including Sparkfun Electronics, also make their own boards that are compatible with Arduino IDE.

Arduino Boards:

There are variety of Arduino board used for different purposes. The board varies in I/O pins, size, etc. The various components present on the Arduino boards are Microcontroller, Digital Input/Output pins, USB Interface and Connector, Analog Pins, Reset Button, Power button, LED's, Crystal Oscillator, and Voltage Regulator. Some components may differ depending on the type of board.

Let's discuss some of the popular Arduino boards.

- ✓ Arduino UNO
- ✓ Arduino Nano
- ✓ Arduino Mega
- ✓ Arduino Due
- ✓ Arduino Bluetooth

Shields:

Shields are defined as the hardware device that can be mounted over the board to increase the capabilities of the projects.

The shield is shown below:



- ❖ The shield together with Arduino can make the projects even smarter and simpler. For example, Ethernet shields are used to connect the Arduino board to the Internet.
- ❖ The shields can be easily attached and detached from the Arduino board. It does not require any complex wiring.

Prerequisite:

The requirement to learn Arduino is the basic knowledge of C and C++ programming language. A basic understanding of circuits, Microcontrollers, and Electronics is also essential.

7.3. Arduino Download:

The Arduino software (IDE) is open-source software. We are required to write the code and upload the code to the board to perform some task.

The Arduino IDE software can be used with any type of Arduino boards. The software is available for various operating system such as, Windows, Linux, and Mac OS X.

The steps to download the Arduino software are listed below:

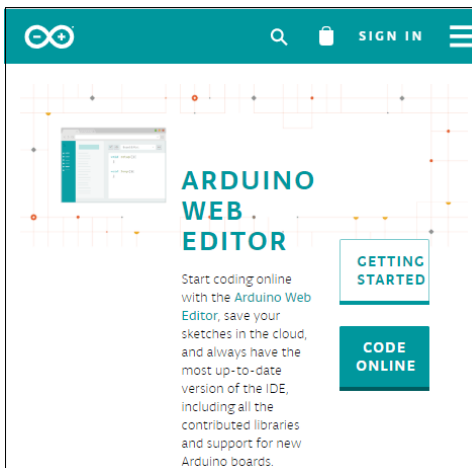
1. Go to the official website of Arduino (<https://www.arduino.cc/>) > Click on SOFTWARE < click on DOWNLOADS, as shown below:



Or

Open the URL <https://www.arduino.cc/en/Main/Software>

2. A page will appear, as shown below:

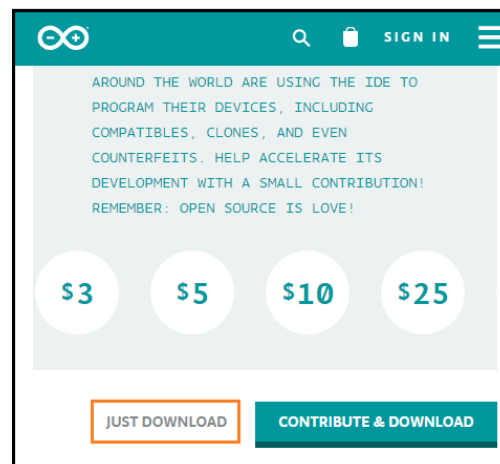


3. Scroll the screen a little, as shown below:

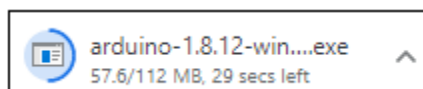


4. Click on the 'Windows Installer' as we are operating with the Windows. We can select the Linux or Mac OS X, accordingly.

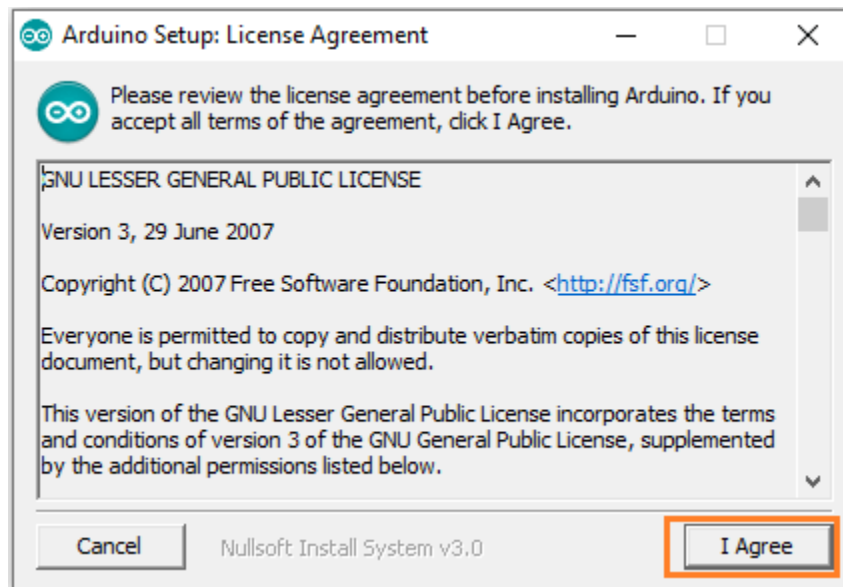
5. A contribution window will appear. We can contribute according to our choice and click on the 'CONTRIBUTE & DOWNLOAD' option. Otherwise, click on the 'JUST DOWNLOAD' option, as shown below:



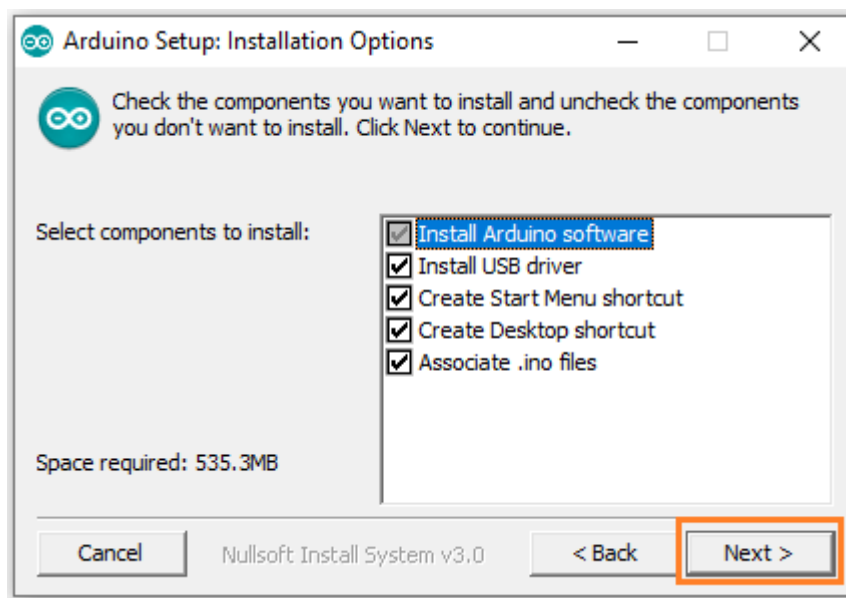
6. The downloading process will start. The downloading file will look like the below image:



7. Wait for few seconds for the downloading process to complete.
8. Open the downloaded file.
9. Grant permission to the Arduino Software on your computer.
10. Accept the license by clicking on 'I Agree' button, as shown below:

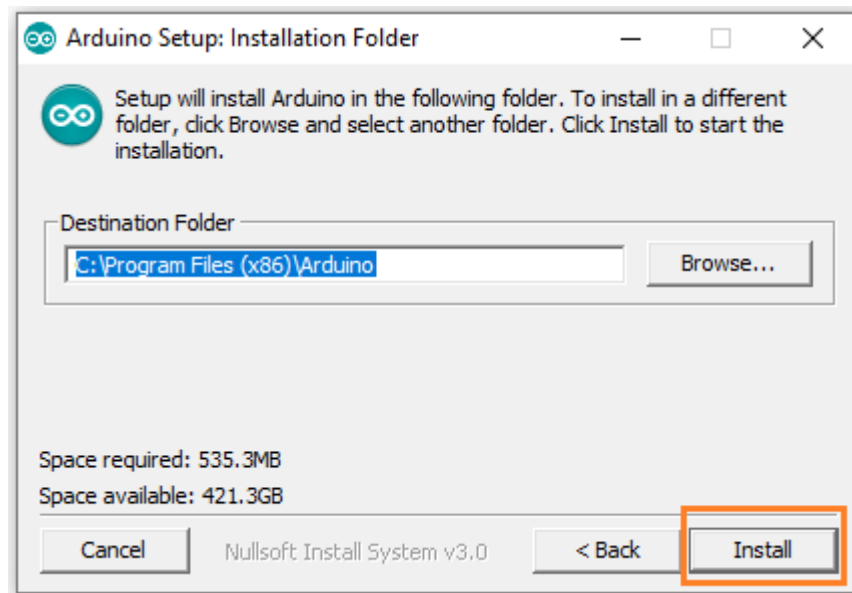


11. Click on the 'Next' button. It is shown below:



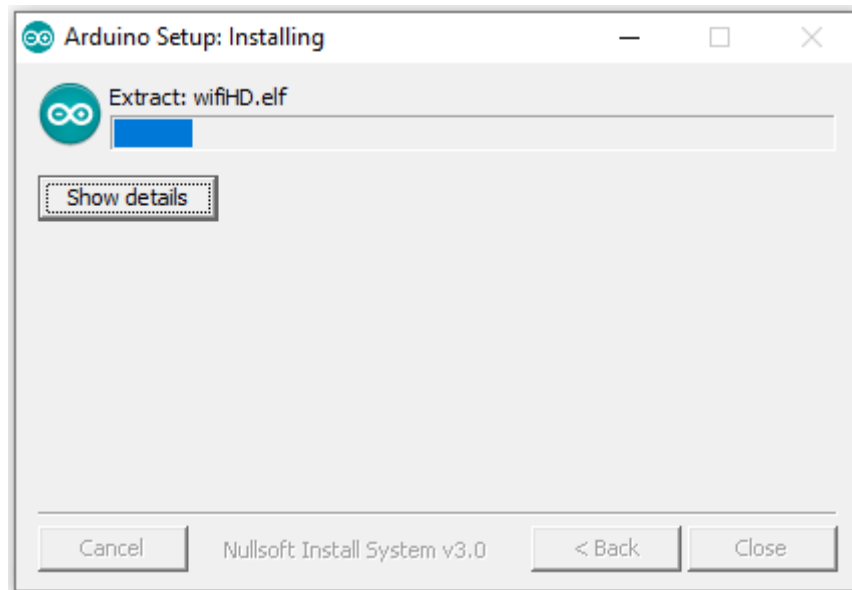
12. The window specifying the location of the installed folder will appear.

Click on the 'Install' button. It is shown below:



If you want to change the location, click on the 'Browse' button.

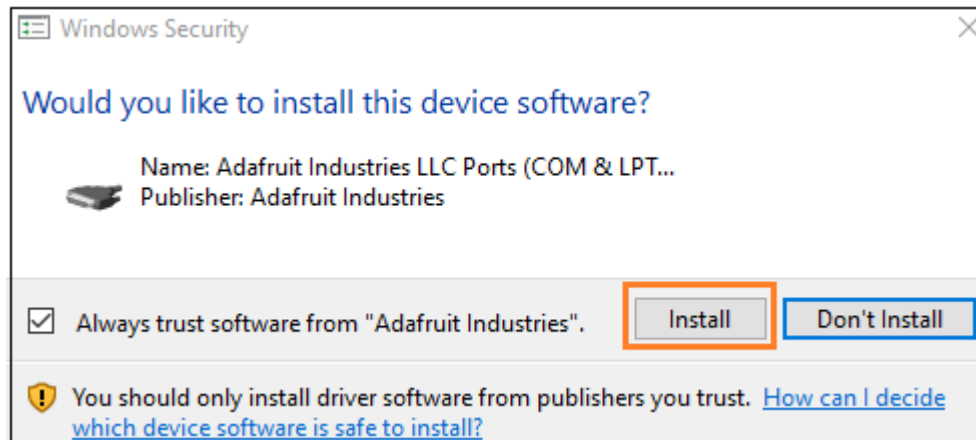
13. The installing process of Arduino will start, as shown below:



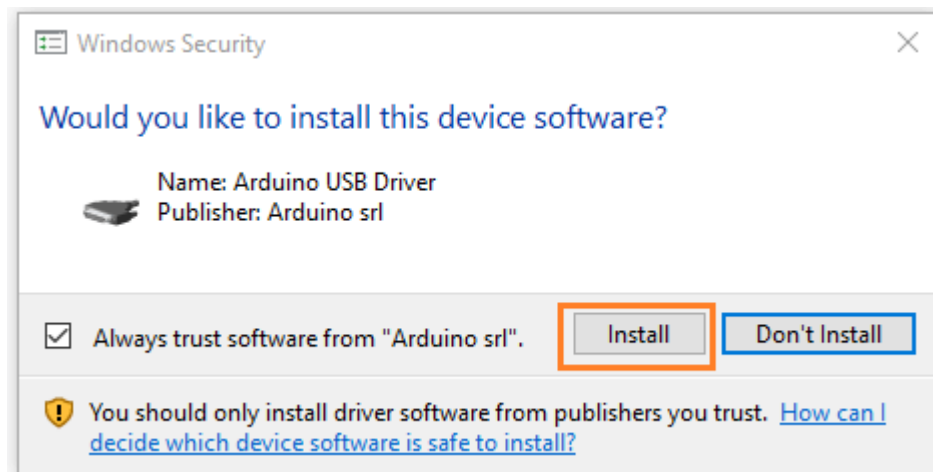
Wait for the installation process to complete.

14. Now, we have to accept the security for the installation. We are required to accept the security Installation three times.

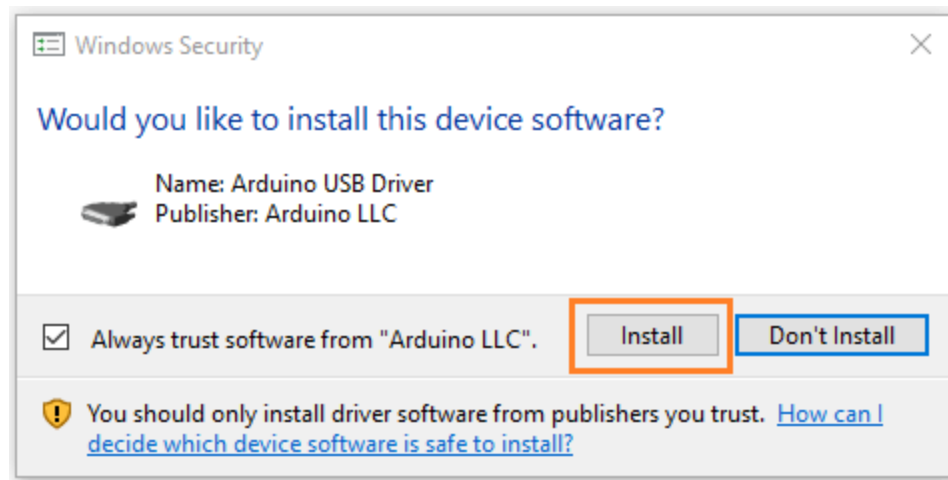
Click on the 'Install' button. The image is shown below:



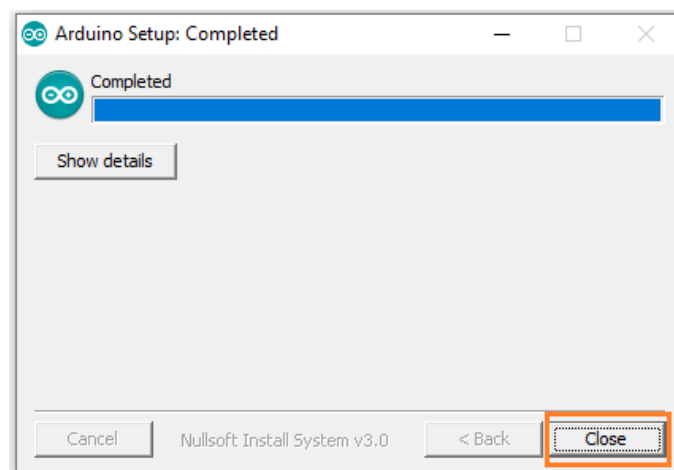
15. Again, click on the 'Install' button. It is shown below:



16. Again, click on the 'Install' button. It is shown below:



17. The installation process is now completed. The window will now appear as:



18. Click on the 'Close' button at the bottom.

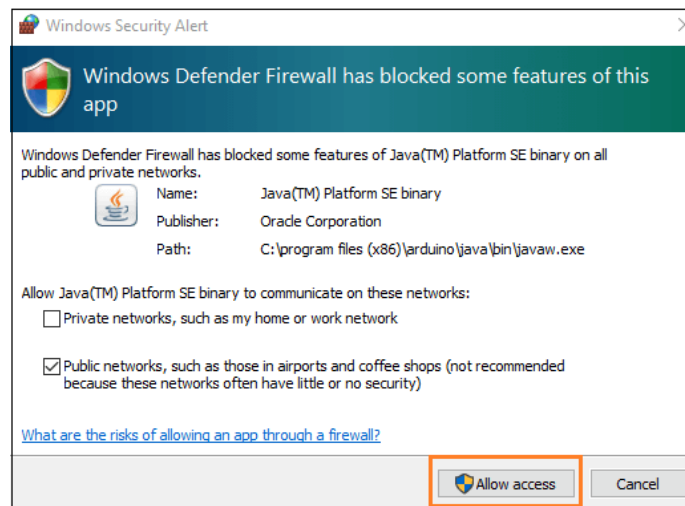
The Arduino IDE software will appear on your desktop, as shown below:



19. Now, open the Arduino software.

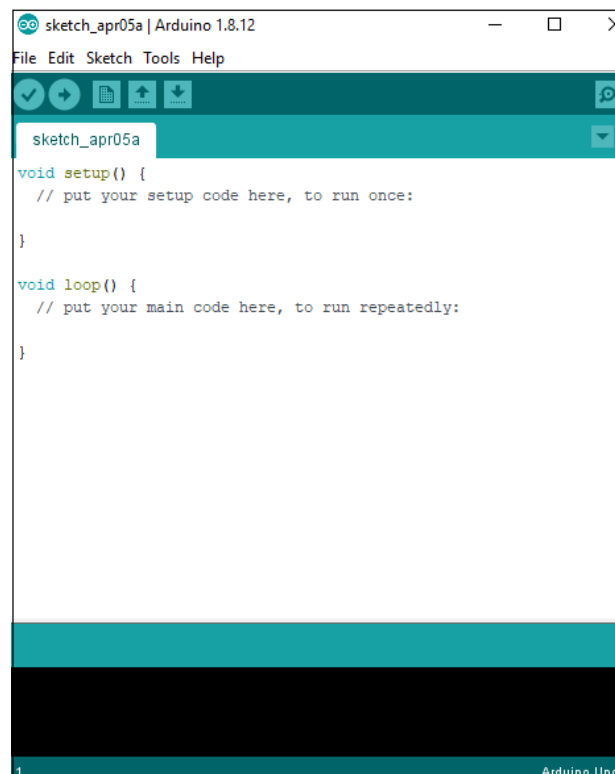
20. The Arduino IDE environment is written in the programming language named as Java. So, we need to allow access to the Java Platform.

As soon we open the Arduino software, a license window will appear, as shown below:



Accept the license by clicking on the 'Allow access' button.

21. The Arduino window will appear as:

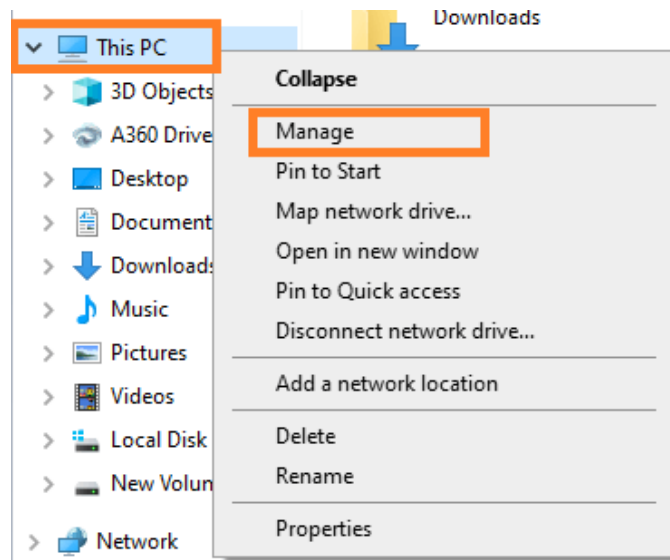


Visibility of the connected Hardware port

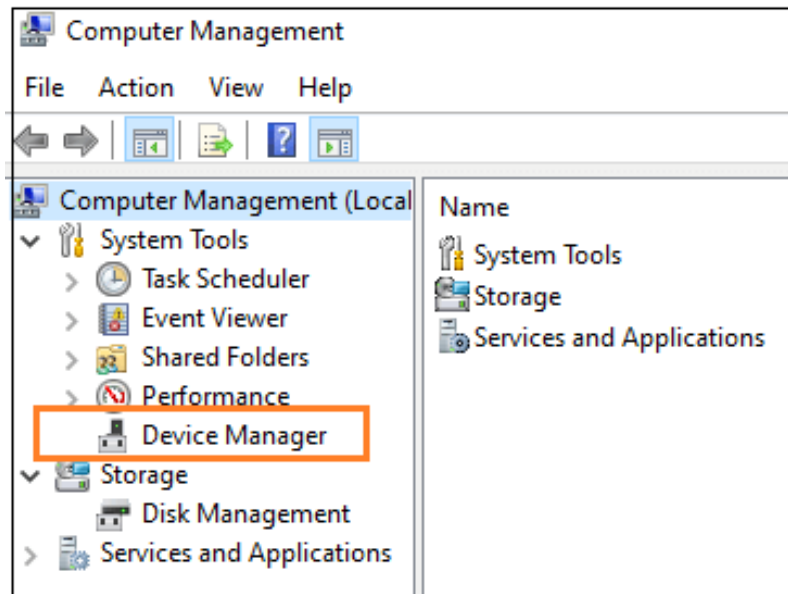
We can view the port of the attached hardware Arduino IDE to our computer.

The steps are listed below:

1. Go to the File Manager and right-click on the This PC option, as shown below:



2. Click on the Manage
3. First, we need to connect the Arduino board to our computer.
4. A window will appear, as shown below:



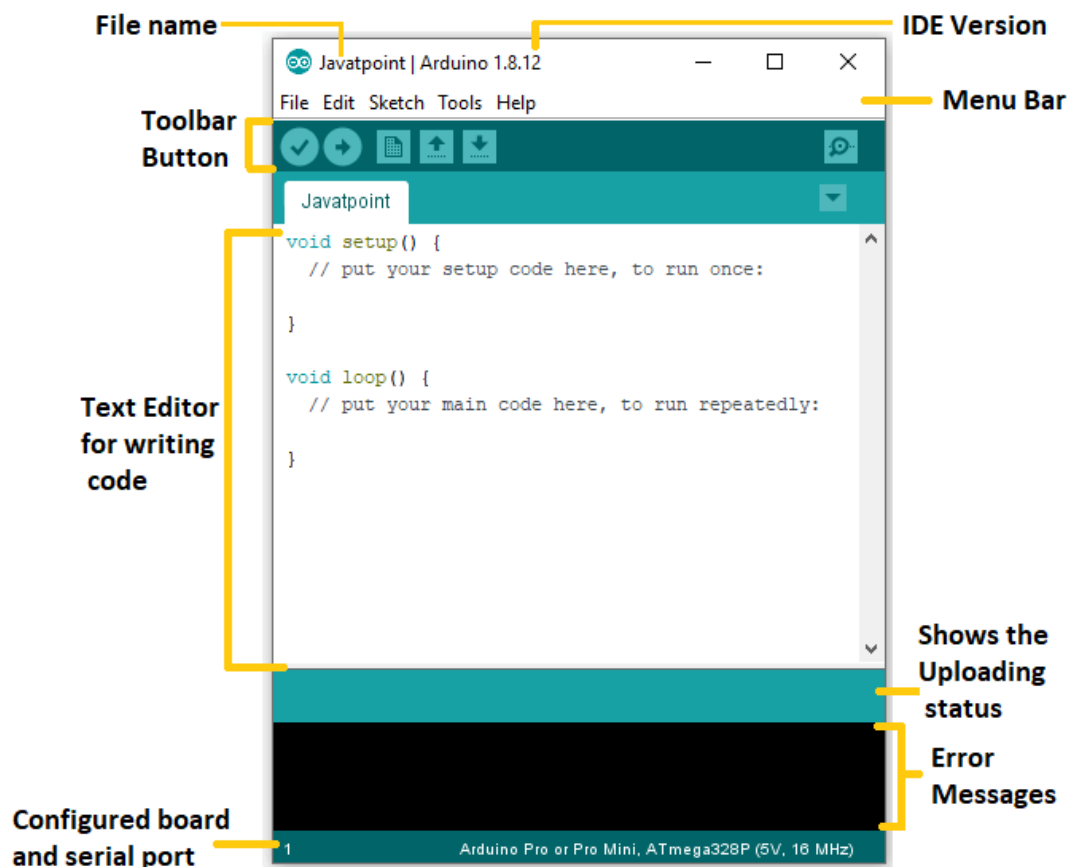
5. Click on the Device Manager
6. Under the PORT option, we can see the ports of the connected hardware.

7.4. Arduino IDE:

The Arduino IDE is an open-source software, which is used to write and upload code to the Arduino boards. The IDE application is suitable for different operating systems such as Windows, Mac OS X, and Linux. It supports the programming languages C and C++. Here, IDE stands for Integrated Development Environment.

The program or code written in the Arduino IDE is often called as sketching. We need to connect the Genuino and Arduino board with the IDE to upload the sketch written in the Arduino IDE software. The sketch is saved with the extension '.ino.'

The Arduino IDE will appear as:



Let's discuss each section of the Arduino IDE display in detail.

Toolbar Button:

The icons displayed on the toolbar are New, Open, Save, Upload, and Verify.

It is shown below:



Upload

The Upload button compiles and runs our code written on the screen. It further uploads the code to the connected board. Before uploading the sketch, we need to make sure that the correct board and ports are selected.

We also need a USB connection to connect the board and the computer. Once all the above measures are done, click on the Upload button present on the toolbar.

The latest Arduino boards can be reset automatically before beginning with Upload. In the older boards, we need to press the Reset button present on it. As soon as the uploading is done successfully, we can notice the blink of the Tx and Rx LED.

If the uploading is failed, it will display the message in the error window.

We do not require any additional hardware to upload our sketch using the Arduino Bootloader. A Bootloader is defined as a small program, which is loaded in the microcontroller present on the board. The LED will blink on PIN 13.

Open

The Open button is used to open the already created file. The selected file will be opened in the current window.

Save

The save button is used to save the current sketch or code.

New

It is used to create a new sketch or opens a new window.

Verify

The Verify button is used to check the compilation error of the sketch or the written code.

Serial Monitor

The serial monitor button is present on the right corner of the toolbar. It opens the serial monitor.

It is shown below:

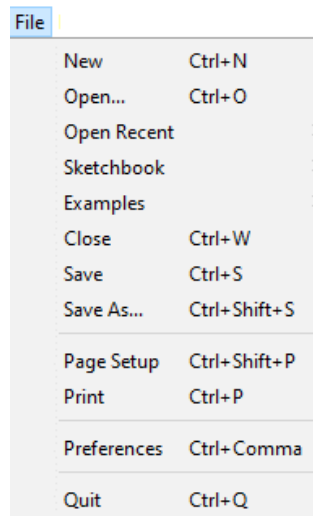


When we connect the serial monitor, the board will reset on the operating system Windows, Linux, and Mac OS X. If we want to process the control characters in our sketch, we need to use an external terminal program. The terminal program should be connected to the COM port, which will be assigned when we connect the board to the computer.

Menu Bar:

1. File Menu:

When we click on the File button on the Menu bar, a drop-down list will appear. It is shown below:



Let's discuss each option in detail.

New

The New button opens the new window. It does not remove the sketch which is already present.

Open

It allows opening the sketch, which can be browsed from the folders and computer drivers.

Open Recent

The Open Recent button contains the list of the recent sketches.

Sketchbook

It stores the current sketches created in the Arduino IDE software. It opens the selected sketch or code in a new editor at an instance.

Examples

It shows the different examples of small projects for a better understanding of the IDE and the board. The IDE provides examples of self-practice.

Close

The Close button closes the window from which the button is clicked.

Save

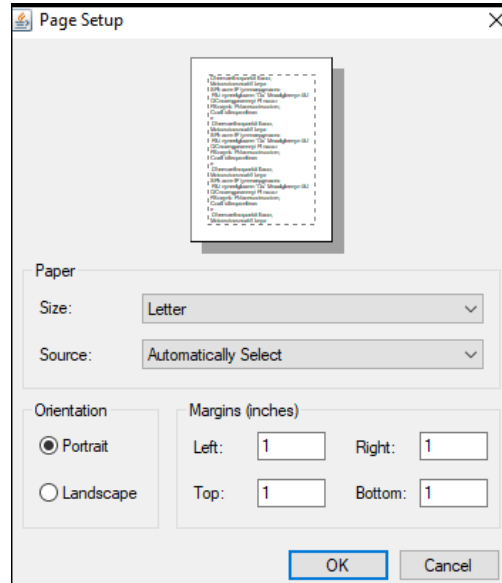
The save button is used to save the current sketch. It also saves the changes made to the current sketch. If we have not specified the name of the file, it will open the 'Save As...' window.

Save As...

We can save the sketch with a different name using the 'Save As...' button. We can also change the name accordingly.

Page Setup

It allows setting the page margins, orientation, and size for printing. The 'Page Setup' window will appear as:



Print

According to the settings specified in the 'Page Setup', it prepares the current sketch for printing.

Preferences

It allows the customization settings of the Arduino IDE.

Quit

The Quit button is used to close all the IDE windows. The same closed sketch will be reopened when we will open the Arduino IDE.

2. Edit Menu:

When we click on the Edit button on the Menu bar, a drop-down list appears. It is shown below:

Edit	
Undo	Ctrl+Z
Redo	Ctrl+Y
Cut	Ctrl+X
Copy	Ctrl+C
Copy for Forum	Ctrl+Shift+C
Copy as HTML	Ctrl+Alt+C
Paste	Ctrl+V
Select All	Ctrl+A
Go to line...	Ctrl+L
Comment/Uncomment	Ctrl+Slash
Increase Indent	Tab
Decrease Indent	Shift+Tab
Increase Font Size	Ctrl+Plus
Decrease Font Size	Ctrl+Minus
Find...	Ctrl+F
Find Next	Ctrl+G
Find Previous	Ctrl+Shift+G

Let's discuss each option in detail.

Undo

The Undo button is used to reverse the last modification done to the sketch while editing.

Redo

The Redo button is used to repeat the last modification done to the sketch while editing.

Cut

It allows us to remove the selected text from the written code. The text is further placed to the clipboard. We can also paste that text anywhere in our sketch.

Copy

It creates a duplicate copy of the selected text. The text is further placed on the clipboard.

Copy for Forum

The 'Copy for Forum' button is used to copy the selected text to the clipboard, which is also suitable for posting to the forum.

Copy as HTML

The 'Copy for Forum' button is used to copy the selected text as HTML to the clipboard. It is desirable for embedding in web pages.

Paste

The Paste button is used to paste the selected text of the clipboard to the specified position of the ***cursor***.

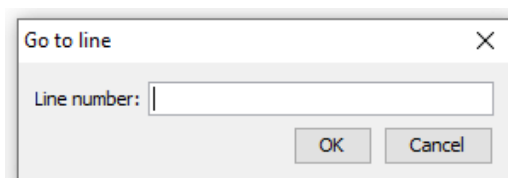
Select All

It selects all the text of the sketch.

Go to line...

It moves the cursor to the specified line number.

The window will appear as:



Comment/De-comment

The Comment/ De-comment button is used to put or remove the comment mark (//) at the beginning of the specified line.

Increase Indent

It is used to add the space at the starting of the specified line. The spacing moves the text towards the right.

Decrease Indent

It is used to subtract or remove the space at the starting of the specified line. The spacing moves the text towards the left.

Increase Font Size

It increases the font size of the written text.

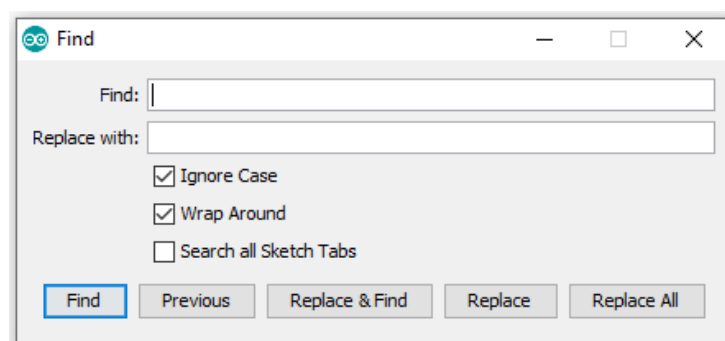
Decrease Font Size

It decreases the font size of the written text.

Find...

It is used to find the specified text. We can also replace the text. It highlights the text in the sketch.

The window will appear as:



Find Next

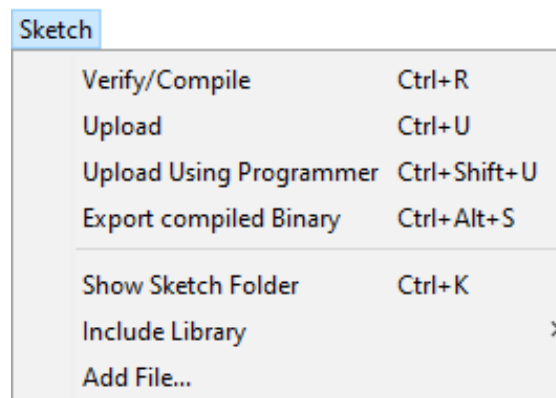
It highlights the next word, which has specified in the 'Find...' window. If there is no such word, it will not show any highlighted text.

Find Previous

It highlights the previous word, which has specified in the 'Find...' window. If there is no such word, it will not show any highlighted text.

3. Sketch Menu:

When we click on the Sketch button on the Menu bar, a drop-down list appears. It is shown below:



Let's discuss each option in detail.

Verify/Compile

It will check for the errors in the code while compiling. The memory in the console area is also reported by the IDE.

Upload

The Upload button is used to configure the code to the specified board through the port.

Upload Using Programmer

It is used to override the Bootloader that is present on the board. We can utilize the full capacity of the Flash memory using the 'Upload Using Programmer' option. To implement this, we need to restore the Bootloader using the Tools-> Burn Bootloader option to upload it to the USB serial port.

Export compiled Binary

It allows saving a .hex file and can be kept archived. Using other tools, .hex file can also be sent to the board.

Show Sketch Folder

It opens the folder of the current code written or sketch.

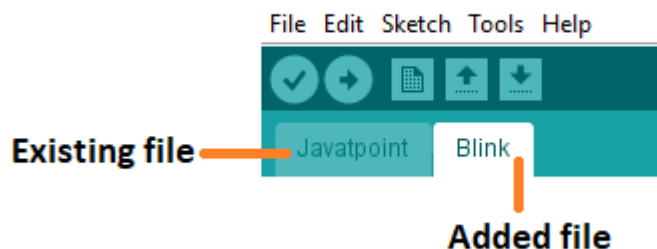
Include Library

Include Library includes various Arduino libraries. The libraries are inserted into our code at the beginning of the code starting with the #. We can also import the libraries from .zip file.

Add File...

The Add File... button is used to add the created file in a new tab on the existing file.

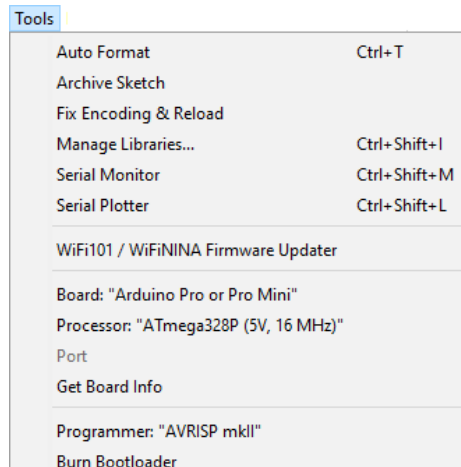
For example, let's add 'Blink' file to the 'Javatpoint' file. The tab will now appear as:



We can also delete the corresponding file from the tab by clicking on the small triangle -> Delete option.

4. Tools Menu:

When we click on the Tools button on the Menu bar, a drop-down list appears. It is shown below:



Let's discuss each option in detail.

Auto Format

The Auto Format button is used to format the written code. For example, lining the open and closed curly brackets in the code.

Archive Sketch

The copy of the current sketch or code is archived in the .zip format. The directory of the archived is same as the sketch.

Fix Encoding and Reload

This button is used to fix the inconsistency between the operating system char maps and editor char map encoding.

Manage Libraries...

It shows the updated list of all the installed libraries. We can also use this option to install a new library into the Arduino IDE.

Serial Monitor

It allows the exchange of data with the connected board on the port.

Serial Plotter

The Serial Plotter button is used to display the serial data in a plot. It comes preinstalled in the Arduino IDE.

WiFi101/WiFiNINA Firmware Updater

It is used to check and update the Wi-Fi Firmware of the connected board.

Board

We are required to select the board from the list of boards. The selected board must be similar to the board connected to the computer.

Processor

It displays the processor according to the selected board. It refreshes every time during the selection of the board.

Port

It consists of the virtual and real serial devices present on our machine.

Get Board Info

It gives the information about the selected board. We need to select the appropriate port before getting information about the board.

Programmer

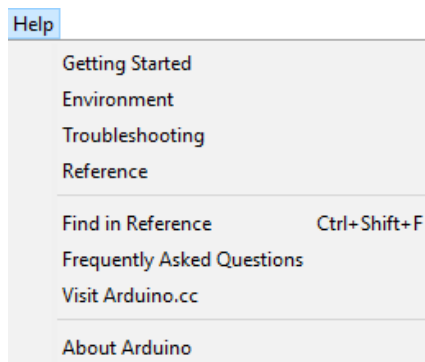
We need to select the hardware programmer while programming the board. It is required when we are not using the onboard USB serial connection. It is also required during the burning of the Bootloader.

Burn Bootloader

The Bootloader is present on the board onto the microcontroller. The option is useful when we have purchased the microcontroller without the bootloader. Before burning the bootloader, we need to make sure about the correct selected board and port.

5. Help Menu:

When we click on the Help button on the Menu bar, a drop-down list will appear. It is shown below:



The Help section includes several documents that are easy to access, which comes along with the Arduino IDE. It consists of the number of options such as Getting Started, Environment, Troubleshooting, Reference, etc. We can also consider the image shown above, which includes all the options under the Help section.

Some documents like Getting started, Reference, etc., can be accessed without the internet connection as well. It will directly link us to the official website of Arduino.

7.5. Arduino Boards:

Arduino is an easy-to-use open platform to create electronics projects. Arduino boards play a vital role in creating different projects. It makes electronics accessible to non-engineers, hobbyists, etc.

The various components present on the Arduino boards are Microcontroller, Digital Input/output pins, USB Interface and Connector, Analog Pins, Reset Button, Power button, LED's, Crystal Oscillator, and Voltage Regulator. Some components may differ depending on the type of board.

The most standard and popular board used over time is Arduino UNO. The ATmega328 Microcontroller present on the UNO board makes it rather powerful than other boards. There are various types of Arduino boards used for different purposes and projects. The Arduino Boards are organized using the Arduino (IDE), which can run on various platforms. Here, IDE stands for Integrated Development Environment.

Let's discuss some common and best Arduino boards.

Types of Arduino Boards

1. Arduino UNO

Arduino UNO is based on an ATmega328P microcontroller. It is easy to use compared to other boards, such as the Arduino Mega board, etc. The Arduino UNO includes 6 analog pin inputs, 14 digital pins, a USB connector, a power jack, and an ICSP (In-Circuit Serial Programming) header.

It is the most used and of standard form from the list of all available Arduino Boards. It is also recommended for beginners as it is easy to use.

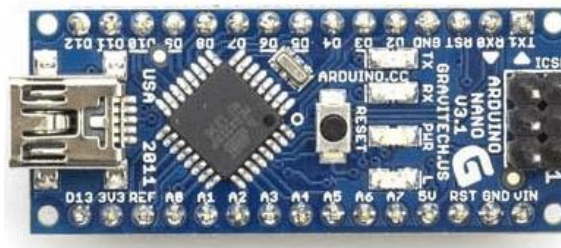


2. Arduino Nano

The Arduino Nano is a small Arduino board based on ATmega328P or ATmega628 Microcontroller. The connectivity is the same as the Arduino UNO board.

The Nano board is defined as a sustainable, small, consistent, and flexible microcontroller board. It is small in size compared to the UNO board. The devices required to start our projects using the Arduino Nano board are Arduino IDE and mini USB.

The Arduino Nano includes an I/O pin set of 14 digital pins and 8 analog pins. It also includes 6 Power pins and 2 Reset pins.



3. Arduino Mega

The Arduino Mega is based on ATmega2560 Microcontroller. The ATmega2560 is an 8-bit microcontroller. We need a simple USB cable to connect to the computer and the AC to DC adapter or battery to get started with it. It has the advantage of working with more memory space.

The Arduino Mega includes 54 I/O digital pins and 16 Analog Input/Output (I/O), ICSP header, a reset button, 4 UART (Universal Asynchronous Receiver/Transmitter) ports, USB connection, and a power jack.



4. Arduino Micro

The Arduino Micro is based on the ATmega32U4 Microcontroller. It consists of 20 sets of pins. The 7 pins from the set are PWM (Pulse Width Modulation) pins, while 12 pins are analog input pins. The other components on board are reset button, 16MHz crystal oscillator, ICSP header, and a micro USB connection.

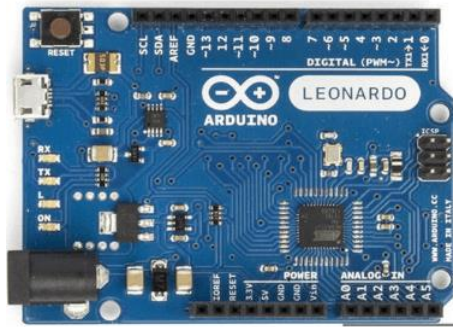
The USB is inbuilt in the Arduino Micro board.



The Arduino Micro is also called as the shrunk version of Arduino Leonardo.

5. Arduino Leonardo

The basic specification of the Arduino Leonardo is the same as the Arduino Micro. It is also based on ATmega32U4 Microcontroller. The components present on the board are 20 analog and digital pins, reset button, 16MHz crystal oscillator, ICSP header, and a micro USB connection.



6. Arduino Due

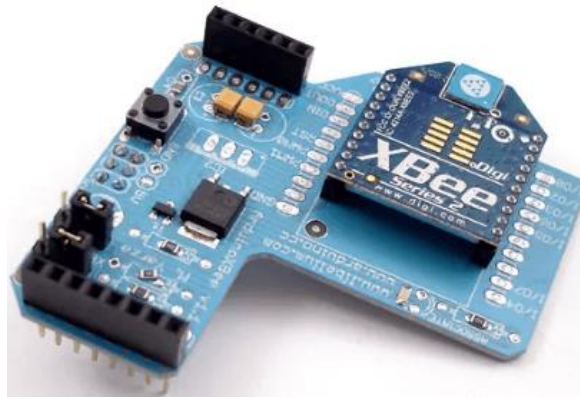
The Arduino Due is based on the 32-bit ARM core. It is the first Arduino board that has developed based on the ARM Microcontroller. It consists of 54 Digital Input/Output pins and 12 Analog pins. The Microcontroller present on the board is the Atmel SAM3X8E ARM Cortex-M3 CPU.



It has two ports, namely, native USB port and Programming port. The micro side of the USB cable should be attached to the programming port.

7. Arduino Shields

The Arduino shields are the boards, which can be plugged on the top of the PCB. The shields further extend the potential of the PCB's. The production of shields is cheap. It is also easy to use. There are various types of Arduino shields that can be used for different purposes. For example, the Xbee shield.

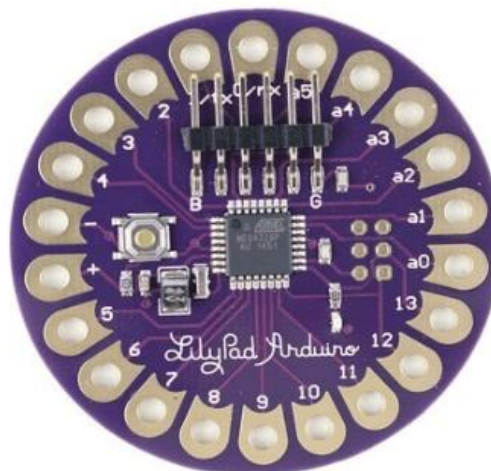


The Xbee shield can be used for wireless communication between multiple Arduino boards over distances upto 300 feet using the Zigbee Module.

8. Arduino Lilypad

The Arduino LilyPad was initially created for wearable projects and e-textiles. It is based on the ATmega168 Microcontroller. The functionality of Lilypad is the same as other Arduino Boards. It is a round, light-weight board with a minimal number of components to keep the size of board small.

The Arduino Lilypad board was designed by Sparkfun and Leah. It was developed by Leah Buechley. It has 9 digital I/O pins.



9. Arduino Bluetooth

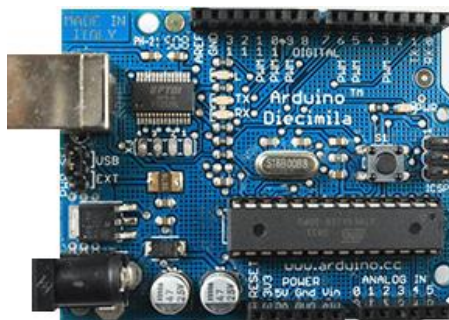
The Arduino Bluetooth board is based on ATmega168 Microcontroller. It is also named as Arduino BT board. The components present on the board are 16 digital pins, 6 analog pins, reset button, 16MHz crystal oscillator, ICSP header, and screw terminals. The screw terminals are used for power.



The Arduino Bluetooth Microcontroller board can be programmed over the Bluetooth as a wireless connection.

10. Arduino Diecimila

The Arduino Diecimila is also based on ATmega628 Microcontroller. The board consists of 6 analog pin inputs, 14 digital Input/Output pins, a USB connector, a power jack, an ICSP (In-Circuit Serial Programming) header, and a reset button.



We can connect the board to the computer using the USB, and can power-on the board with the help of AC to DC adapter. The Diecimila was initially developed to mark the 10000 delivered boards of Arduino. Here, Diecimila means 10,000 in Italian.

11. Arduino Robot

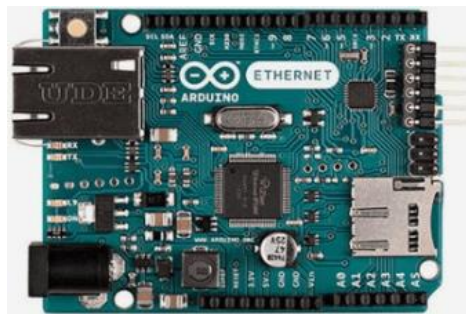
The Arduino Robot is called as the tiny computer. It is widely used in robotics. The board comprises of the speaker, five-button, color screen, two motors, an SD card reader, a digital compass, two potentiometers, and five floor sensors.



The Robot Library can be used to control the actuators and the sensors.

12. Arduino Ethernet

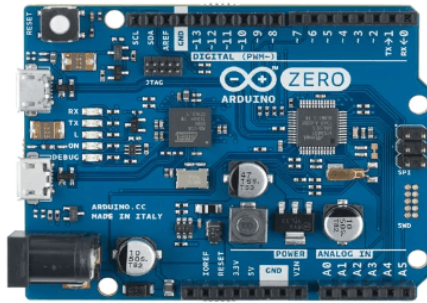
The Arduino Ethernet is based on the ATmega328 Microcontroller. The board consists of 6 analog pins, 14 digital I/O pins, crystal oscillator, reset button, ICSP header, a power jack, and an RJ45 connection.



With the help of the Ethernet shield, we can connect our Arduino board to the internet.

13. Arduino Zero

The Arduino Zero is generally called as the 32-bit extension of the Arduino UNO. It is based on ATmel's SAM21 MCU. The board consists of 6 analog pin inputs, 14 digital Input/Output pins, a USB connector, a power jack, and an ICSP (In-Circuit Serial Programming) header, UART port pins, a power header, and AREF button.



The Embedded debugger of Atmel is also supported by the Arduino Zero. The function of Debugger is to provide a full debug interface, which does not require additional hardware.

14. Arduino Esplora

The Arduino Esplora boards allow easy interfacing of sensors and actuators. The outputs and inputs connected on the Esplora board make it unique from other types of Arduino boards. The board includes outputs, inputs, a small microcontroller, a microphone, a sensor, a joystick, an accelerometer, a temperature sensor, four buttons, and a slider.



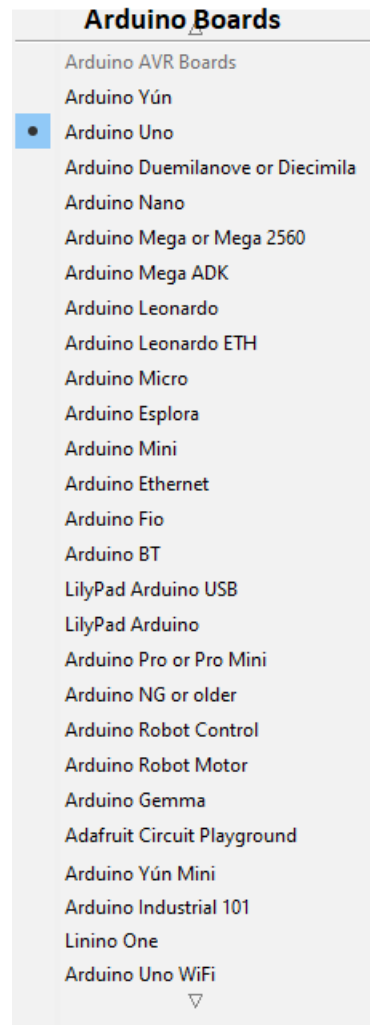
15. Arduino Pro Micro

The structure of Arduino Pro Micro is similar to the Arduino Mini board, except the Microcontroller ATmega32U4. The board consists of 12 digital Input/output pins, 5 PWM (Pulse Width Modulation) pins, Tx and Rx serial connections, and 10-bit ADC (Analog to Digital Converter).



List of available boards in Arduino software

The list of boards that we can see in the Arduino software is shown below:



Here, the dot represents the select Arduino board in the Arduino IDE.

7.6. Arduino UNO:

The Arduino UNO is a standard board of Arduino. Here UNO means 'one' in Italian. It was named as UNO to label the first release of Arduino Software. It was also the first USB board released by Arduino. It is considered as the powerful board used in various projects. Arduino.cc developed the Arduino UNO board.

Arduino UNO is based on an ATmega328P microcontroller. It is easy to use compared to other boards, such as the Arduino Mega board, etc. The board consists of digital and analog Input/Output pins (I/O), shields, and other circuits.

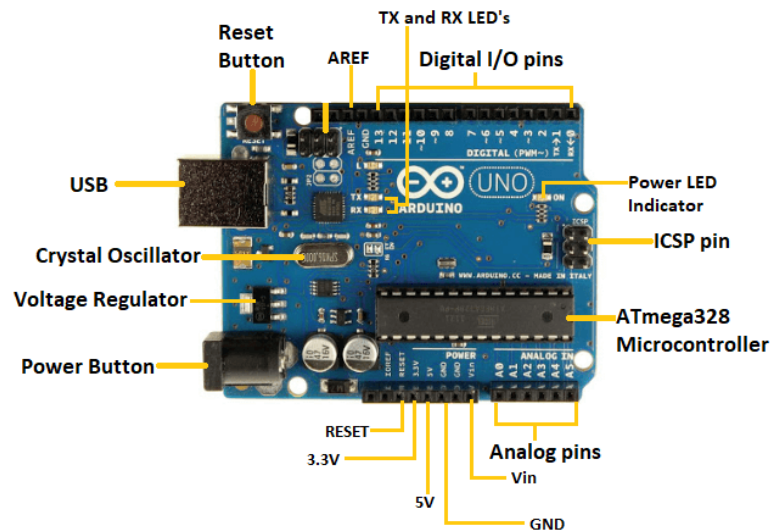
The Arduino UNO includes 6 analog pin inputs, 14 digital pins, a USB connector, a power jack, and an ICSP (In-Circuit Serial Programming) header. It is programmed based on IDE, which stands for Integrated Development Environment. It can run on both online and offline platforms.

The IDE is common to all available boards of Arduino.

The Arduino board is shown below:



The components of Arduino UNO board are shown below:



Let's discuss each component in detail.

- ❖ ATmega328 Microcontroller- It is a single chip Microcontroller of the ATmel family. The processor code inside it is of 8-bit. It combines Memory (SRAM, EEPROM, and Flash), Analog to Digital Converter, SPI serial ports, I/O lines, registers, timer, external and internal interrupts, and oscillator.
- ❖ ICSP pin - The In-Circuit Serial Programming pin allows the user to program using the firmware of the Arduino board.
- ❖ Power LED Indicator- The ON status of LED shows the power is activated. When the power is OFF, the LED will not light up.
- ❖ Digital I/O pins- The digital pins have the value HIGH or LOW. The pins numbered from D0 to D13 are digital pins.
- ❖ TX and RX LED's- The successful flow of data is represented by the lighting of these LED's.
- ❖ AREF- The Analog Reference (AREF) pin is used to feed a reference voltage to the Arduino UNO board from the external power supply.
- ❖ Reset button- It is used to add a Reset button to the connection.
- ❖ USB- It allows the board to connect to the computer. It is essential for the programming of the Arduino UNO board.
- ❖ Crystal Oscillator- The Crystal oscillator has a frequency of 16MHz, which makes the Arduino UNO a powerful board.

- ❖ Voltage Regulator- The voltage regulator converts the input voltage to 5V.
- ❖ GND- Ground pins. The ground pin acts as a pin with zero voltage.
- ❖ Vin- It is the input voltage.
- ❖ Analog Pins- The pins numbered from A0 to A5 are analog pins. The function of Analog pins is to read the analog sensor used in the connection. It can also act as GPIO (General Purpose Input Output) pins.

Why is Arduino recommended over other boards for beginners?

The USB port in the Arduino board is used to connect the board to the computer using the USB cable. The cable acts as a serial port and as the power supply to interface the board. Such dual functioning makes it unique to recommend and easy to use for beginners.

What is the main difference between Arduino UNO and Arduino Nano?

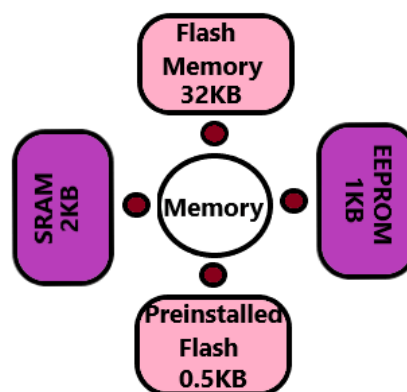
The Arduino Nano has a compact size and mini USB cable than the Arduino UNO.

What is the main difference between Arduino UNO and Arduino Mega?

The Arduino UNO is a standard board recommended to beginners, while Arduino Mega is used for complex projects due to its greater memory space.

Memory:

The memory structure is shown in the below image:



The preinstalled flash has a bootloader, which takes the memory of 0.5 Kb.

Here, SRAM stands for Static Random Access Memory, and EEPROM stands for Electrically Erasable Programmable Read-Only Memory.

Technical Specifications of Arduino UNO:

The technical specifications of the Arduino UNO are listed below:

- ❖ There are 20 Input/Output pins present on the Arduino UNO board. These 20 pins include 6 PWM pins, 6 analog pins, and 8 digital I/O pins.
- ❖ The PWM pins are Pulse Width Modulation capable pins.
- ❖ The crystal oscillator present in Arduino UNO comes with a frequency of 16MHz.
- ❖ It also has an Arduino integrated WiFi module. Such Arduino UNO board is based on the Integrated WiFi ESP8266 Module and ATmega328P microcontroller.
- ❖ The input voltage of the UNO board varies from 7V to 20V.
- ❖ Arduino UNO automatically draws power from the external power supply. It can also draw power from the USB.

How to get started with Arduino UNO?

We can program the Arduino UNO using the Arduino IDE. The Arduino IDE is the Integrated Development program, which is common to all the boards.

We can also use Arduino Web Editor, which allows us to upload sketches and write the code from our web browser (Google Chrome recommended) to any Arduino Board. It is an online platform.

The USB connection is essential to connect the computer with the board. After the connection, the PWR pins will light in green. It is a green power LED.

The steps to get started with Arduino UNO are listed below:

- ✓ Install the drivers of the board.

As soon we connect the board to the computer, Windows from XP to 10 will automatically install the board drivers.

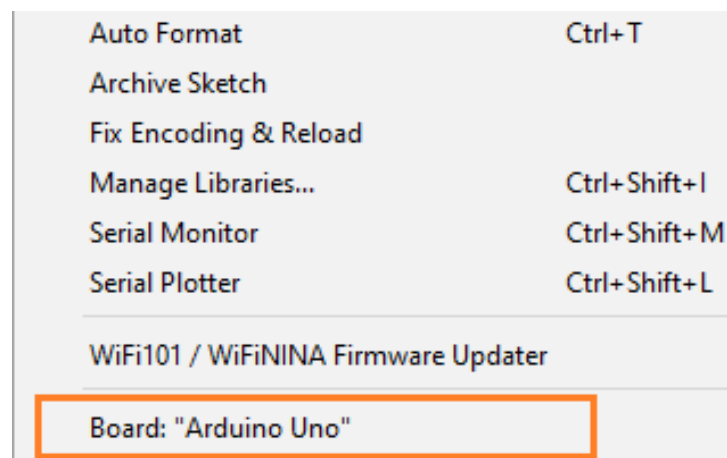
But, if you have expanded or downloaded the zip package, follow the below steps:

1. Click on Start -> Control Panel -> System and Security.
2. Click on System -> Device Manager -> Ports (COM & LPT) -> Arduino UNO (COMxx).
If the COM & LPT is absent, look Other Devices -> Unknown Device.
3. Right-click to Arduino UNO (COMxx) -> Update Driver Software -> Browse my computer for driver software.
4. Select the file ".inf" to navigate else, select "ArduinoUNO.inf" .
5. Installation Finished.

Open the code or sketch written in the Arduino software.

Select the type of board.

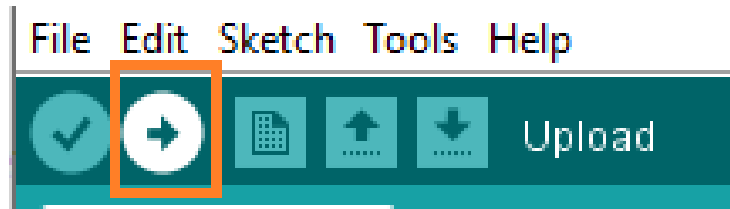
Click on 'Tools' and select Board, as shown below:



Select the port. Click on the Tools -> Port (select the port). The port likely will be COM3 or higher. For example, COM6, etc. The COM1 and COM2 ports will not appear, because these two ports are reserved for the hardware serial ports.

Now, upload and run the written code or sketch.

To upload and run, click on the button present on the top panel of the Arduino display, as shown below:



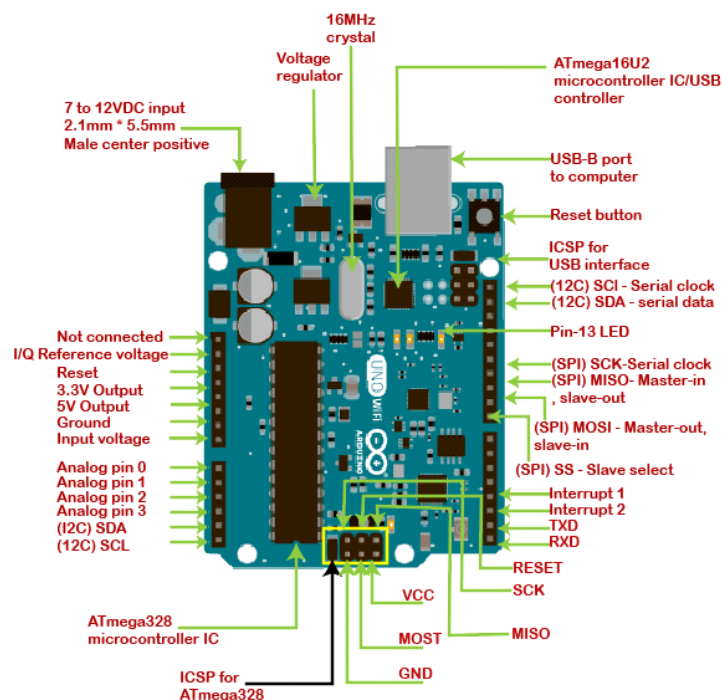
Within the few seconds after the compile and run of code or sketch, the RX and TX light present on the Arduino board will flash.

The 'Done Uploading' message will appear after the code is successfully uploaded. The message will be visible in the status bar.

Arduino UNO Pinout

The Arduino UNO is a standard board of Arduino, which is based on an ATmega328P microcontroller. It is easier to use than other types of Arduino Boards.

The Arduino UNO Board, with the specification of pins, is shown below:



Let's discuss each pin in detail.

✓ **ATmega328 Microcontroller**

It is a single chip Microcontroller of the ATmel family. The processor core inside it is of 8-bit. It is a low-cost, low powered, and a simple microcontroller. The Arduino UNO and Nano models are based on the ATmega328 Microcontroller.

✓ **Voltage Regulator**

The voltage regulator converts the input voltage to 5V. The primary function of voltage regulator is to regulate the voltage level in the Arduino board. For any changes in the input voltage of the regulator, the output voltage is constant and steady.

✓ **GND**

Ground pins. The ground pins are used to ground the circuit.

✓ **TXD and RXD**

TXD and RXD pins are used for serial communication. The TXD is used for transmitting the data, and RXD is used for receiving the data. It also represents the successful flow of data.

✓ **USB Interface**

The USB Interface is used to plug-in the USB cable. It allows the board to connect to the computer. It is essential for the programming of the Arduino UNO board.

✓ **RESET**

It is used to add a Reset button to the connection.

✓ **SCK**

It stands for Serial Clock. These are the clock pulses, which are used to synchronize the transmission of data.

✓ **MISO**

It stands for Master Input/ Slave Output. The save line in the MISO pin is used to send the data to the master.

✓ **VCC**

It is the modulated DC supply voltage, which is used to regulate the IC's used in the connection. It is also called as the primary voltage for IC's present on the Arduino board. The Vcc voltage value can be negative or positive with respect to the GND pin.

✓ **Crystal Oscillator**

The Crystal oscillator has a frequency of 16MHz, which makes the Arduino UNO a powerful board.

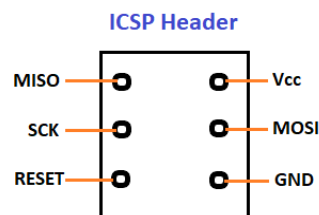
✓ **ICSP**

It stands for In-Circuit Serial Programming. The users can program the Arduino board's firmware using the ICSP pins.

The program or firmware with the advanced functionalities is received by microcontroller with the help of the ICSP header.

The ICSP header consists of 6 pins.

The structure of the ICSP header is shown



below:

It is the top view of the ICSP header.

➤ **SDA**

It stands for Serial Data. It is a line used by the slave and master to send and receive data. It is called as a data line, while SCL is called as a clock line.

➤ SCL

It stands for Serial Clock. It is defined as the line that carries the clock data. It is used to synchronize the transfer of data between the two devices. The Serial Clock is generated by the device and it is called as master.

➤ SPI

It stands for Serial Peripheral Interface. It is popularly used by the microcontrollers to communicate with one or more peripheral devices quickly. It uses conductors for data receiving, data sending, synchronization, and device selection (for communication).

➤ MOSI

It stands for Master Output/ Slave Input.

The MOSI and SCK are driven by the Master.

✓ SS

It stands for Slave Select. It is the Slave Select line, which is used by the master. It acts as the enable line.

✓ I2C

It is the two-wire serial communication protocol. It stands for Inter Integrated Circuits. The I2C is a serial communication protocol that uses SCL (Serial Clock) and SDA (Serial Data) to receive and send data between two devices.

3.3V and 5V are the operating voltages of the board.

7.7. Arduino Projects:

Project - 01

Blinking an LED

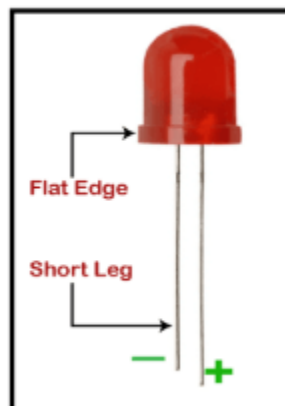
It is the simple basic project created using Arduino. LED (Light Emitting Diode) is an electronic device, which emits light when the current passes through its terminals. LED's are used in various applications. It is also used as an ON/OFF indicator in different electronic devices.

In this project, we will connect the LED to the digital pin on the Arduino board. The LED will work as a simple light that can be turned ON and OFF for a specified duration.

Structure of LED

An LED is a two-terminal device. The two terminals are called as Cathode and Anode.

It is shown below:



The long terminal is called Anode, and the shorter terminal is called Cathode. Here, cathode is the negative terminal and anode is the positive terminal.

Components of the project

The components used in the blinking of an LED are listed below:

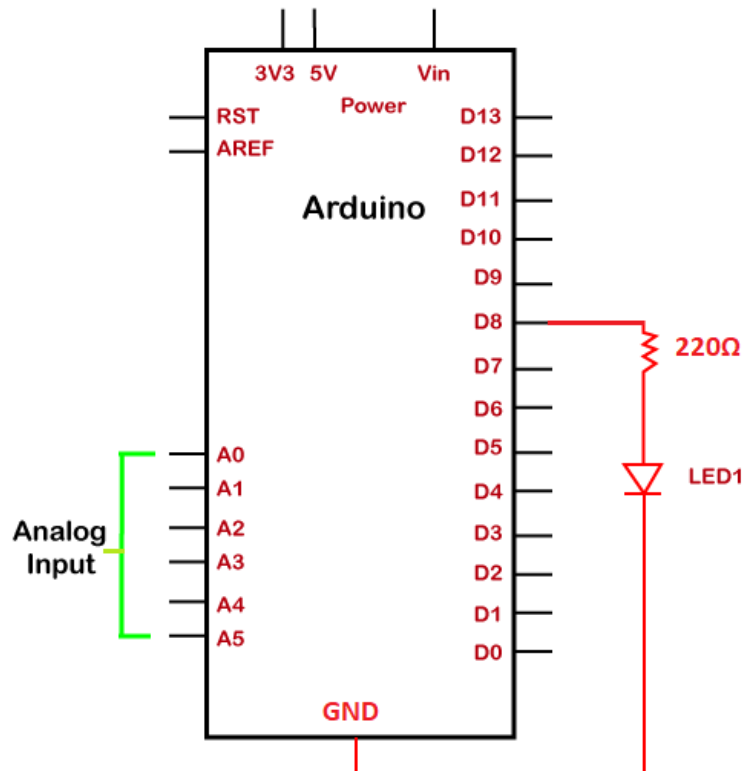
1. 1 x Arduino UNO board.
2. We can use any version of the UNO board, such as UNO R3, etc. We can also use other types of Arduino boards, such as Arduino Zero, Arduino Micro, etc.
3. 1 x Breadboard
4. 2 x Jump wires
5. 1 x LED
6. 1 x Resistor of 220 Ohm.

We can use a resistor of any value upto 470 Ohms. We can use other value of resistors as well, depending on our circuit requirements. Usually, the value should not exceed the allowable forward current.

Structure of the project

The structure clearly shows the pinout of the UNO board. It also displays the LED and resistance connected to the board.

It is shown below:



Sketch

We need to install the Arduino IDE, to begin with the coding, which is already discussed.

Open the IDE and start with the coding, which is given below:

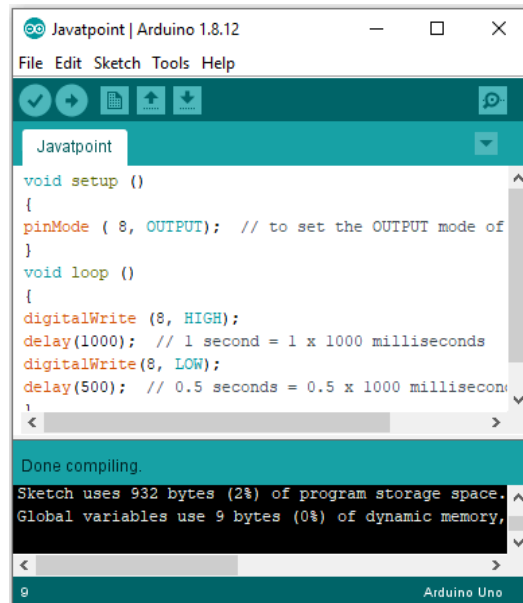
```
1. void setup ()
2. {
3.   pinMode ( 8, OUTPUT); // to set the OUTPUT mode of pin number 8.
4. }
5. void loop ()
6. {
7.   digitalWrite (8, HIGH);
8.   delay(1000); // 1 second = 1 x 1000 milliseconds
9.   digitalWrite (8, LOW);
10.  delay(500); // 0.5 second = 0.5 x 1000 milliseconds
11. }
```

We can modify the delay duration according to our choice or as per the requirements.

Every statement of coding is explained in Arduino coding basics. You can open the URL for clear understanding.

Note: Make sure the code is free of errors.

The sketch will be uploaded to the board after the correct compiling, as shown below:



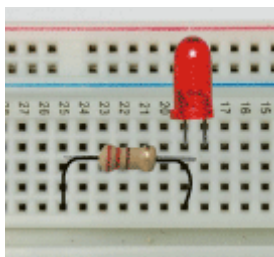
We are required to click on the Verify button to compile the code.

The RX and TX LED on the board will light up after the successful uploading of the code.

Procedure

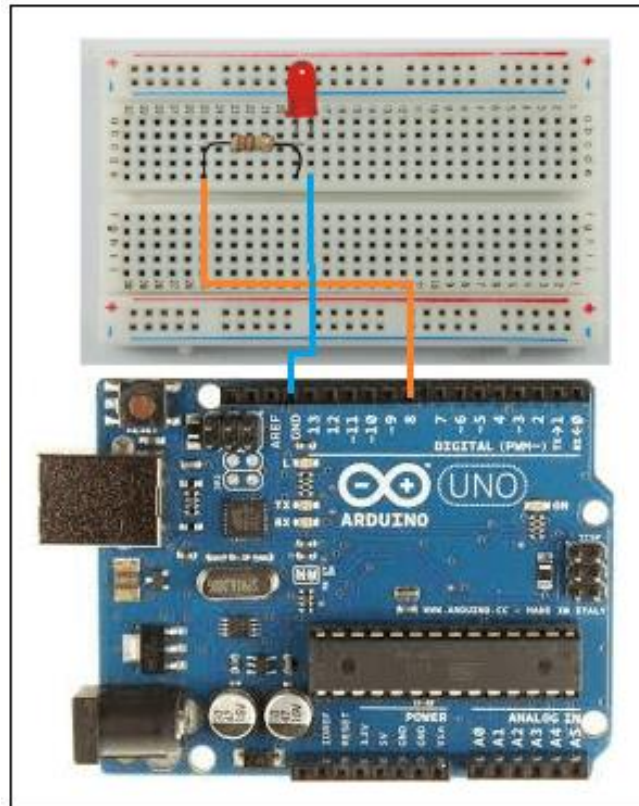
The procedure to join the components of the project is shown below:

- ❖ Attach an LED on the breadboard. We need to plug-in the two terminals of an LED into the holes of the breadboard.
- ❖ We can plug-in the LED anywhere on the breadboard.
- ❖ Connect the resistor in series with the LED, as shown below:



- ❖ Connect the left leg of the resistor (connected in series with red LED) to the digital output pin of the UNO board, i.e., PIN 8.

- ❖ Connect the negative/shorter terminal (Cathode) of the LED to the GND pin of the UNO board using the wire, as shown below:



Here, the orange wire is connected to the PIN 8, and the blue wire is connected to the GND.

The shorter terminal indicates the negative. So, we will connect the shorter terminal to the Ground (GND).

- ❖ Connect the USB cable.
- ❖ Select the board and serial port in the Arduino IDE.
- ❖ Upload the sketch or code on the board.
- ❖ The LED will dim and light for the specified duration.

Important points

The important points to be considered in this project are listed below:

- ✓ The resistor must be connected in series with the LED.

The resistor prevents the excess current from reaching the LED. The excess current in the connection can burn the LED. Hence, a resistor in series with the LED is used in the connection.

- ✓ We can use any pin as the OUTPUT pin. For example, 8, 13, 7, 4, and 3. The other pins are PWM and analog pins.
- ✓ One terminal of the LED is connected to the Ground while the other terminal is connected to the digital pin. The digital pin has only two values 0 or 1.
 - HIGH = 1
 - LOW = 0
- ✓ Arduino UNO board is recommended for all basic projects because it is easy to understand and implement. It is also the standard Arduino board from all types of boards used. It supplies power and also acts as a serial port.

Project - 02

Blinking Two LED

We have already discussed a project of blinking an LED. Here, we will discuss a project of blinking two LED's.

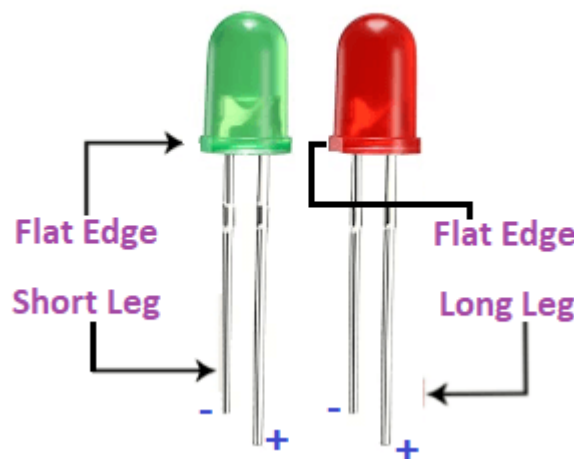
The concept of blinking two LED's is similar to the blinking of a single LED. As we know, we can use the resistance of any value, so let's take the resistors of 470 Ohms. The resistors reduce the amount of current reaching the LED, which saves the LED from being burnt.

We can also use other resistors depending on the circuit limit and requirements.

Let's start with the project.

Structure of two LED's

The structure of red and green LED is shown below:



The long terminal is called Anode (positive charged), and the short terminal is called Cathode (negative charged).

Components

The components used in the project are listed below:

1. 1 x Arduino UNO board. We can also use other types of Arduino boards, such as Arduino Mega, Arduino Micro, etc.
2. 1 x Breadboard
3. 4 x Jump wires
4. 1 x Red LED
5. 1 x Green LED, We need to take 2 LEDs of any color. Here, we will use the red and green color LED.
6. 2 x Resistor of 470 Ohm.

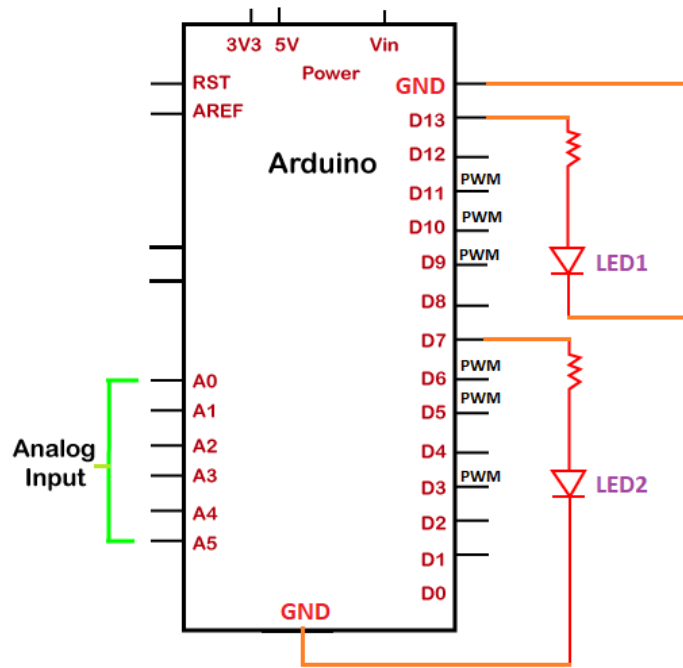
Structure of the project

Here, we will use the digital output pin number 13 and 7. The positive terminal of the red LED is connected to the PIN 13, and the negative terminal (anode) is connected to the ground.

Similarly, the positive terminal (cathode) of the green LED is connected to PIN 7 and the negative terminal is connected to the ground.

As mentioned, two resistors each of 470 Ohms, will be connected in series to the two LEDs in the project.

The structure will represent the pinout diagram of the project. It is shown below:



Sketch

Open the Arduino IDE (Integrated Development Environment) and start with the coding, which is given below:

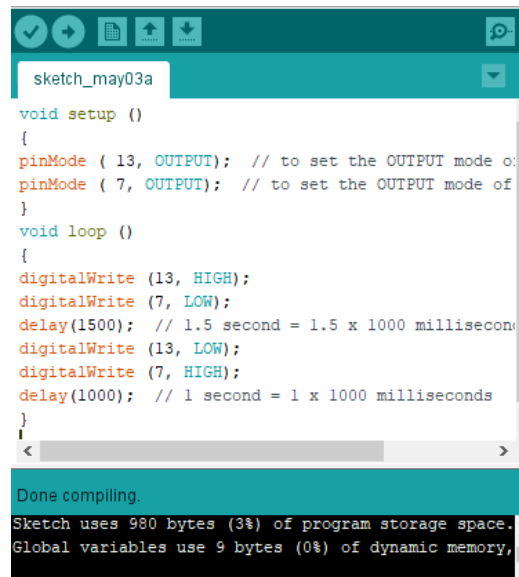
```

1. void setup ()
2. {
3.   pinMode ( 13, OUTPUT); // to set the OUTPUT mode of pin number 13.
4.   pinMode ( 7, OUTPUT); // to set the OUTPUT mode of pin number 7.
5. }
6. void loop ()
7. {
8.   digitalWrite (13, HIGH);
9.   digitalWrite (7, LOW);
10.    delay(1500); // 1.5 second = 1.5 x 1000 milliseconds
11.    digitalWrite (13, LOW);
12.    digitalWrite (7, HIGH);
13.    delay(1000); // 1 second = 1 x 1000 milliseconds
14.    }

```

We can modify the delay duration according to our choice or as per the requirements.

The sketch will be uploaded to the board after the correct compiling, as shown below:



```
sketch_may03a
void setup ()
{
  pinMode ( 13, OUTPUT); // to set the OUTPUT mode of
  pinMode ( 7, OUTPUT); // to set the OUTPUT mode of
}
void loop ()
{
  digitalWrite (13, HIGH);
  digitalWrite (7, LOW);
  delay(1500); // 1.5 second = 1.5 x 1000 milliseconds
  digitalWrite (13, LOW);
  digitalWrite (7, HIGH);
  delay(1000); // 1 second = 1 x 1000 milliseconds
}
Done compiling.
Sketch uses 980 bytes (3%) of program storage space.
Global variables use 9 bytes (0%) of dynamic memory,
```

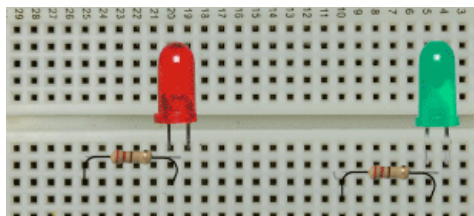
Click on the Verify button present on the toolbar to compile the code.

The RX and TX LED on the board will light up after the successful uploading of the code.

Procedure

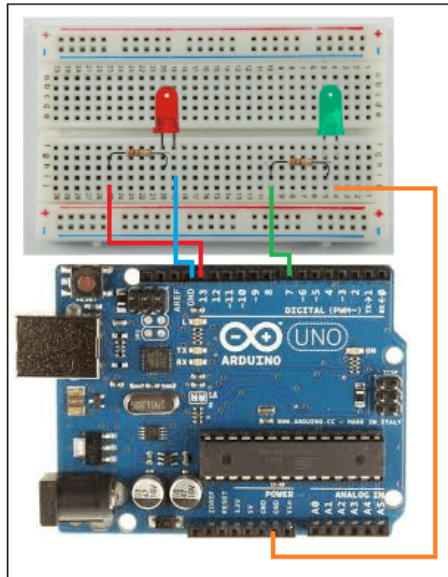
The procedure to join the components of the project is shown below:

- ❖ Plug-in the two LED adjacent to each other on the breadboard.
- ❖ Now, plug-in the resistors of 470 Ohm in series with the two LED, as shown below:



We need to check that the plug-in is performed correctly, as shown above. For any confusion, check the pin diagram shown above in the heading- Structure of project.

- ❖ Connect the left leg of the resistor (connected in series with red LED) to the digital output pin of the UNO board, i.e., PIN 13.
- ❖ Connect the left leg of the resistor (connected in series with green LED) to the digital output pin of the UNO board, i.e., PIN 7.
- ❖ Connect the negative/shorter terminal (Cathode) of the red and green LED to the GND pin of the UNO board using the wire, as shown below:



Here, the red wire is connected to the PIN 13, and the blue wire is connected to the GND.

Similarly, the green wire is connected to the PIN 7, and the orange wire is connected to the GND.

The shorter terminal indicates the ground. So, we will connect the shorter terminal to the Ground (GND).

- ❖ Connect the USB cable.
- ❖ Select the board and serial port in the Arduino IDE.
- ❖ Upload the sketch or code on the board.
- ❖ The LED will dim and light for the specified duration. Here, the green and red LED will light alternatively.

It means when the red LED will be ON, the green LED will be OFF and vice versa.

Project -03

Blinking various LEDs using Arrays

We have already discussed how to blink a single LED, two LEDs, and LEDs using a loop, in previous topics.

Here, we will discuss a project to blink five LEDs using array. All the five LEDs will light one after the other.

Hardware Required

The components required for the project are listed below:

- 5 x red LED
- 5 x 220 Ohm Resistors
- Arduino UNO R3 board
- Jump wires

We can use any color LED as per our choice.

Principle

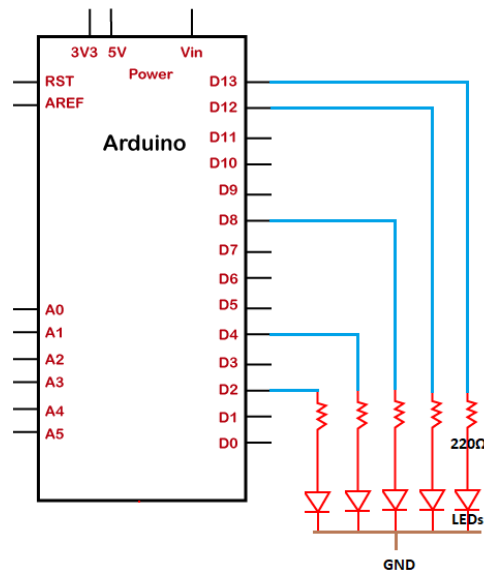
We will connect the five LEDs to pins 13, 12, 8, 4, and 2 of the Arduino board. The required resistance of the resistors is enough to light up an LED without damaging the board and other components.

The LED arranged one after another will light up. We can also change or rearrange the LEDs connected to the specified pin number on the board.

Structure of the project

The structure clearly shows the UNO board's pinout, and the five LEDs with resistors in series are connected to the board.

It is shown below:



Connection

The connection of the above project is discussed below:

- ❖ Connect the resistor of 220 Ohm in series with the five LEDs. Now connect it to the pin number 13, 12, 8, 4, and 2 of the Arduino board.
- ❖ Connect the negative terminal of the five LEDs to the GND (Ground).

Sketch

The code to light the five LEDs is given below:

```

1. int timer = 500;
2. int LEDPins[] = { 13, 12, 8, 4, 2};    // an array of declared pin numbers on the b
oard
3. int countOFpin = 6;        // the number of arrays
4. void setup()
5. {
6.    // we have declared an array to intialize the LED pins as OUTPUT
7.    for (int PIN = 0; PIN < countOFpin; PIN= PIN + 1)
8.    {
9.        pinMode(LEDPins[PIN], OUTPUT);
10.    }

```

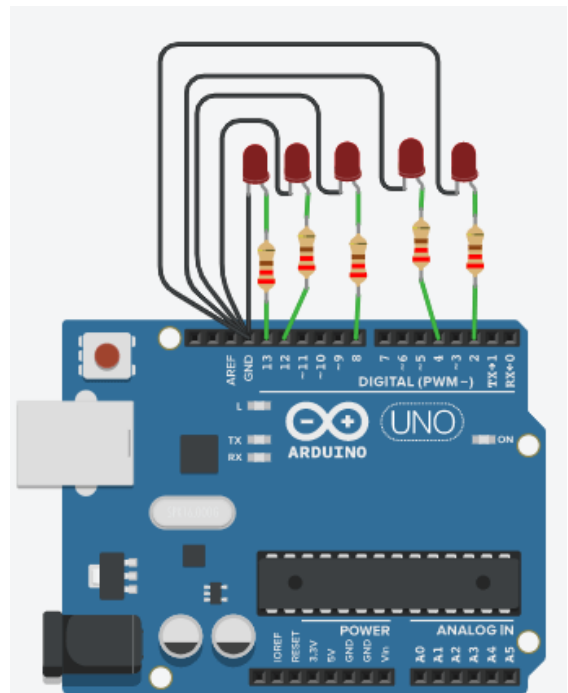
```

11. }
12. void loop()
13. {
14. // loop starting from the lowest pin in the array to the highest:
15. for (int PIN = 0; PIN < countOFpin; PIN++) {
16. // turns the pin ON:
17. digitalWrite(LEDpins[PIN], HIGH);
18. delay(timer);
19. // turnS the pin OFF:
20. digitalWrite(LEDpins[PIN], LOW);
21. }
22. // loop from the highest pin in the array to the lowest:
23. // It means the LEDs will light in the reverse direction as used above
24. for (int PIN = countOFpin - 1; PIN >= 0; PIN-- )
25. {
26. digitalWrite(LEDpins[PIN], HIGH);
27. delay(timer);
28. digitalWrite(LEDpins[PIN], LO
W);
29. // We can also specify the time i
inside the delay( ) instead of the delcaring t
he timer
30. }
31. }

```

Connection Diagram

We will show the connection using the Simulator because the connections become clearer and more precise.



We can make the same connection using the hardware devices.

Project - 04

Arduino LDR

Arduino LDR (Light Dependent Resistor) project uses the photoresistor to light an LED. The LED will light up whenever there is dark or no light over the sensor.

What is photoresistor?

It is defined as a light-controlled resistor, which is also called as LDR. It is a variable resistor that controls the resistance in accordance with the received light intensity. It means, the resistance decreases as intensity of light increases.

Let's start with the project.

Hardware Required

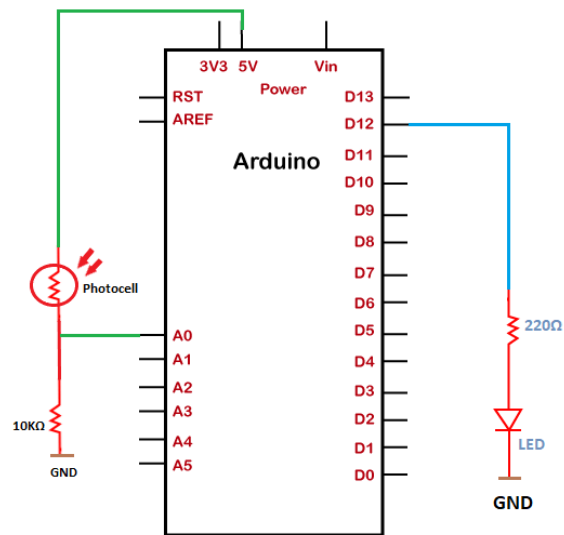
The components required for the project are listed below:

- 1 x red LED
- 1 x 220 Ohm Resistor
- 1 x 10K Ohm Resistor
- Arduino UNO R3 board
- Jump wires
- 1 x photoresistor

We can use any color LED as per our choice.

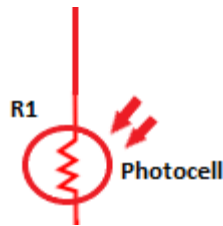
Structure of the project

The structure of the project is shown below:



How to calculate the output voltage using photoresistor?

The structure of photoresistor is shown below:



The formula to calculate the dark output voltage is given below:

$$V_{out} = V_{in} \cdot \frac{R2}{R1 + R2}$$

where,

$V_{in} = 5V$

V_{out} = Output voltage

Here, $R2$ is the resistance connected in series with the photoresistor = 10K Ohm.

$R1$ is the resistance of the photoresistor.

Note: The resistance decreases with increasing light. Hence, the output voltage will increase. It means that the output voltage calculated at the light will be higher than the output voltage calculated at dark.

Here, dark and light specify the light falling on the sensor.

Connection

The connection of the above project is discussed below:

- ❖ Connect the positive leg of the LED in series with the resistor to the pin number 12 of the Arduino board.
- ❖ Connect the negative leg of the LED to the Ground.
- ❖ Connect one edge of the photoresistor to the 5V pin on the Arduino board.
- ❖ Attach a 10K ohm resistance in series with another edge of the photoresistor and connect it to the GND.
- ❖ Connect the edge of the photoresistor to the analog pin A0.

Note: We have connected the LED only to enhance the project. We can also create the project without using the LED. It will not impact the output.

Sketch

Consider the below code:

```
1. const int LEDpin = 12;
2. const int photoPIN = A0;
3. void setup()
4. {
5.   // initializing the serial communication:
6.   Serial.begin(9600);
7.   pinMode(photoPIN, INPUT);
8.   pinMode(LEDpin, OUTPUT);
9. }
10. void loop() {
```

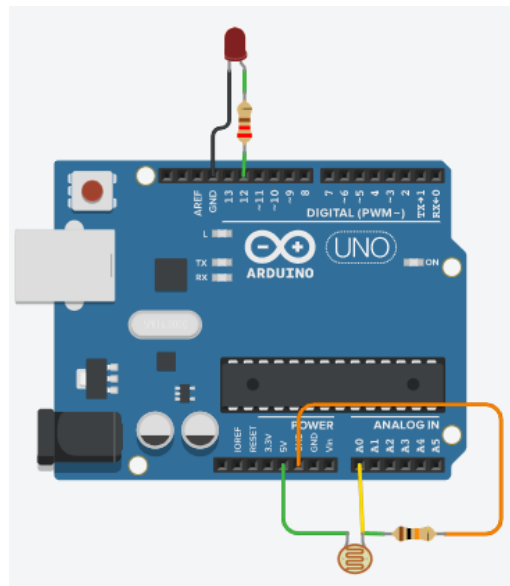
```

11. // read the sensor:
12. int sensorStatus = analogRead(photoPIN);
13. // now, it will check the reading or status of the sensor is < 200
14. // if it is, LED will be HIGH
15. if (sensorStatus <200)
16. {
17.   digitalWrite(LEDpin, HIGH); // LED is ON
18.   Serial.println(" LED is ON, status of sensor is DARK");
19. }
20. else
21. {
22.   digitalWrite(LEDpin, LOW);
23.   Serial.println(" *****");
24. }
25. }

```

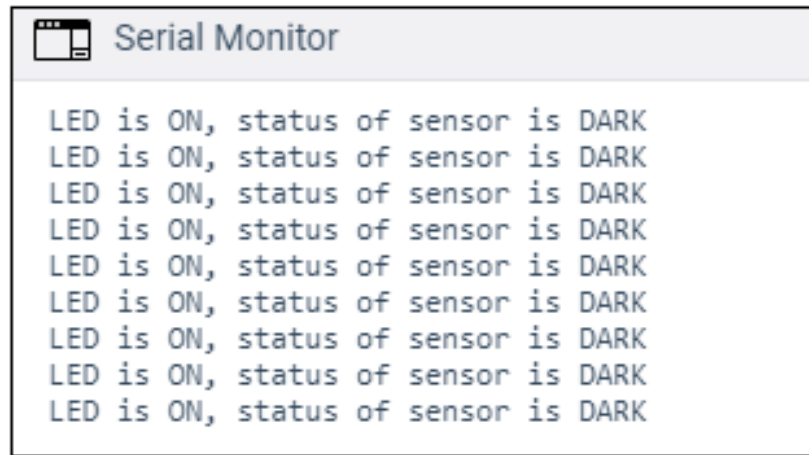
Connection Diagram

The connection diagram is shown below:



Output

The output on the serial monitor will appear as:

A screenshot of a 'Serial Monitor' window. The window has a title bar with a small icon and the text 'Serial Monitor'. The main area contains ten lines of text, each reading 'LED is ON, status of sensor is DARK'.

```
Serial Monitor  
LED is ON, status of sensor is DARK  
LED is ON, status of sensor is DARK  
LED is ON, status of sensor is DARK  
LED is ON, status of sensor is DARK  
LED is ON, status of sensor is DARK  
LED is ON, status of sensor is DARK  
LED is ON, status of sensor is DARK  
LED is ON, status of sensor is DARK  
LED is ON, status of sensor is DARK
```

LED is one because there is dark over the sensor.

Project - 05

Arduino Ultrasonic distance sensor

The Ultrasonic sensor or HC-SRO4 is used to measure the distance of the object using SONAR.

It emits the Ultrasound at a frequency of 40KHZ or 40000 Hz. The frequency travels through the air and strikes the object on its path. The rays bounce back from the object and reach back to the module.

The four terminals of HC-SRO4 are VCC, TRIG, ECHO, and GND. The voltage supply or VCC is +5V. We can connect the ECHO and TRIG terminal to any of the digital I/O pin on the specific Arduino board.

The Ultrasonic sensors work best for medium ranges.

The resolution is 0.3cm.

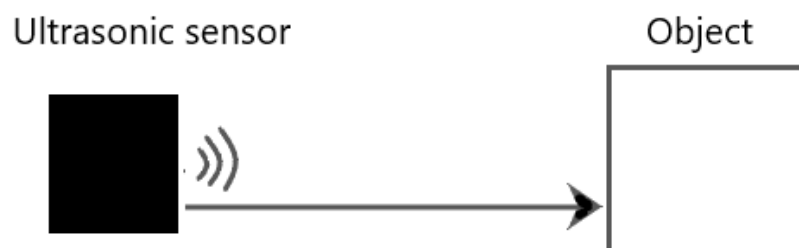
The medium ranges of the sensor are 10cm to 3m. It works best at this duration.

The maximum range the sensor may detect is 4.5m.

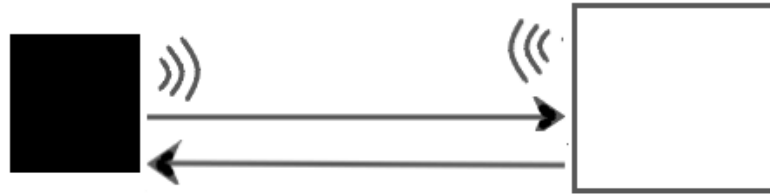
How does Ultrasonic sensor work?

Let's understand how the sensor works.

- ❖ It sends ultra-high frequency samples.



- ❖ When samples strike the object, it bounces back from the object.



- ❖ The distance sensor reports the time it takes between the sending and receiving of the samples.

Example

Let's consider an example.

An object is 40cm away from the Ultrasonic sensor. The speed of sound in air is 340m/s. We need to calculate the time (in Microseconds).

Solution:

$$v = 340\text{m/s} = 0.034\text{cm/us (centimeter/microseconds)}$$

$$\text{time} = \text{distance/speed}$$

$$\text{time} = 40/0.034$$

$$\text{time} = 1176 \text{ microseconds}$$

The speed of sound from the echo pin will double because the wave travels forward and backward (bounces).

So, to calculate the distance, we need to divide it by 2. It is shown below:

$$\text{distance} = \text{time} \times \text{speed of sound}/2$$

$$\text{distance} = \text{time} \times 0.034/2$$

Structure of Ultrasonic Sensor

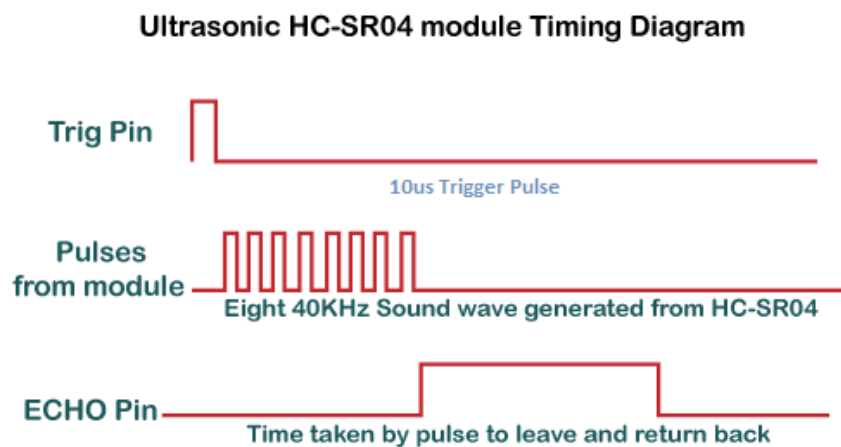
The structure of HC-SRO4 is shown below:



Ultrasonic sensor Timing Diagram

We will set the TRIG pin to HIGH for some time (about 3 to 100 microseconds). As soon the TRIG pin is LOW, the Ultrasonic sensor sends the pulses and sets the ECHO pin to HIGH. When the sensor gets the reflected pulses, it sets the ECHO pin to LOW. We need to measure the time for which the ECHO pin was HIGH.

The timing diagram of the ultrasonic sensor HC-SRO4 is shown below:



Let's start creating the Arduino ultrasonic sensor to measure distance.

Hardware Required

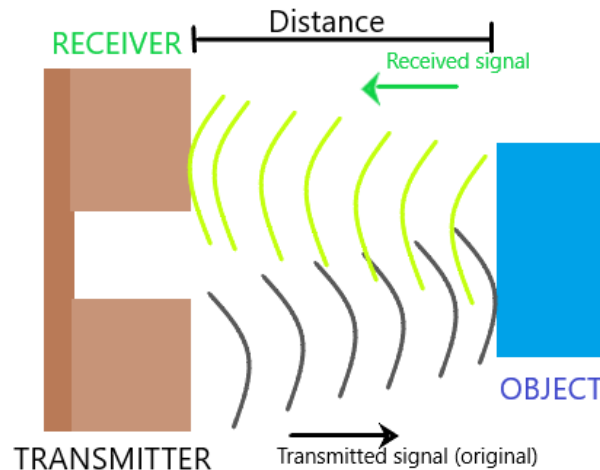
The components required to create the project are listed below:

- Arduino UNO R3 board (We can also use any Arduino board)
- Jump wires
- Ultrasonic sensor HC-SR04
- Breadboard

Principle

We need to first set the TRIG (triggered) pin at HIGH. It will send out the burst of 8 cycles called the sonic burst, which will travel at the sound speed. It will be further received by

the ECHO pin. The time traveled by the sound wave is considered the ECHO pin's output time in microseconds.



We will use the `PulseIn()` function to read the time from the output of the ECHO pin. It will wait for the specified pin to go HIGH and LOW. The function would return the timing at the end.

The TRIG pin is set LOW for 4 microseconds and then HIGH for 15 microseconds.

The timing will be calculated in microseconds.

Procedure

The steps to connect the Ultrasonic sensor to the board are listed below:

- ❖ Connect the VCC pin of HC-SRO4 to 5V of the Arduino board.
- ❖ Connect the GND pin of HC-SRO4 to GND of the Arduino board.
- ❖ Connect the TRIG pin of HC-SRO4 to pin 6 of the Arduino board.
- ❖ Connect the ECHO pin of HC-SRO4 to pin 5 of the Arduino board.

Sketch

Consider the below code:

1. `#define ECHOPin 5` // it defines the ECHO pin of the sensor to pin 5 of Arduino

```

2. #define TRIGpin 6
3. // we have defined the variable
4. long duration; // variable for the duration of sound wave travel
5. int distance; // variable for the distance measurement
6. void setup()
7. {
8.   pinMode(TRIGpin, OUTPUT); // It sets the ECHO pin as OUTPUT
9.   pinMode(ECHOpin, INPUT); // It sets the TRIG pin as INPUT
10.    Serial.begin(9600); // // Serial Communication at the rate of 9600 bps
11.    Serial.println("Test of the Ultrasonic Sensor HC-
    SR04"); // It will appear on Serial Monitor
12.    Serial.println("with the Arduino UNO R3 board");
13.  }
14.  void loop()
15.  {
16.    // It first sets the TRIG pin at LOW for 2 microseconds
17.    digitalWrite(TRIGpin, LOW);
18.    delayMicroseconds(4);
19.    // It now sets TRIG pin at HIGH for 15 microseconds
20.    digitalWrite(TRIGpin, HIGH);
21.    delayMicroseconds(15);
22.    digitalWrite(TRIGpin, LOW);
23.    // It will read the ECHO pin and will return the time
24.    duration = pulseIn(ECHOpin, HIGH);
25.    // distance formula
26.    distance = duration*(0.034/2); // (speed in microseconds)
27.    // Speed of sound wave (340 m/s)divided by 2 (forward and backward bounce)
28.    // To display the distance on Serial Monitor
29.    Serial.print("Distance: ");
30.    Serial.print(distance);
31.    Serial.println(" cm"); //specified unit of distance
32.  }

```

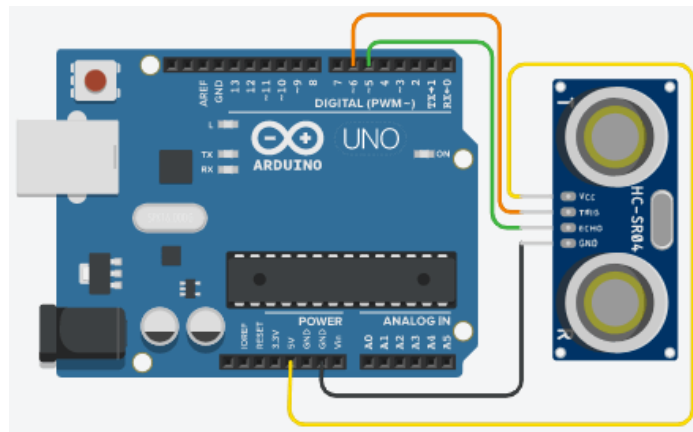
Steps to upload the code to the project

The steps are listed below:

- ❖ Open the Arduino IDE.
- ❖ Select the type of board from Tools -> Board -> Arduino UNO.
- ❖ Select the port from Tools -> Port -> COM..
- ❖ Upload the sketch to the connection diagram.

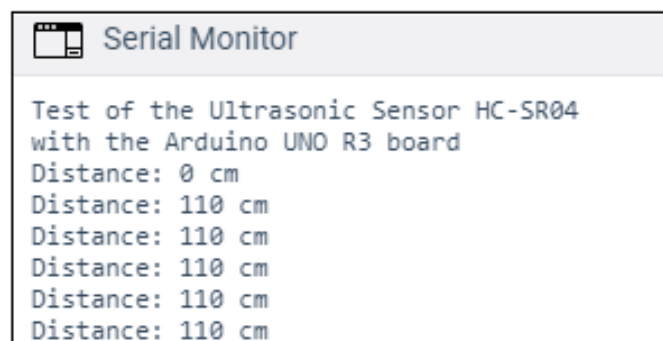
Connection Diagram

The connection diagram is shown below:



Output

The output on the serial monitor will appear as:



Project - 06

Arduino Ultrasonic Range finder

In previous topic of Arduino Ultrasonic Distance Sensor, we have used a four-terminal Ultrasonic sensor.

Here, we will use a three-terminal Ultrasonic sensor, which is shown below:



It has three terminal GND (Ground), 5V, and SIG (signal). The process and the function of this sensor is similar to the Ultrasonic Distance Sensor.

The frequency waves travel through the air and strike the object on its path. The waves bounce back from the object and reach back to the module.

The Ultrasonic Range sensor pings the obstacles or objects with the ultrasound.

It detects the range from 3 cm or 4m or 400 cm.

The example of such a sensor is:

SEN136B5B

It is a sensor from SeedStudio. We need to switch the state between HIGH and LOW to notice the output.

Let's start the project.

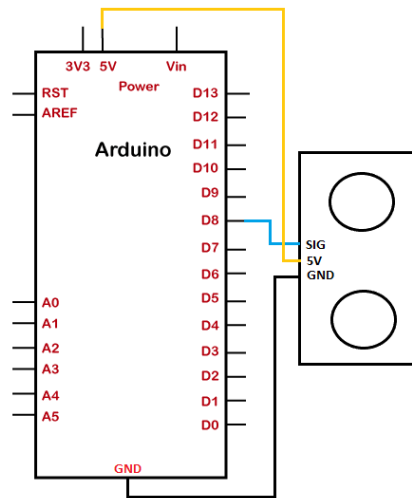
Hardware Required

The components required to create the project are listed below:

- ❖ Arduino UNO R3 board (We can also use any Arduino board)
- ❖ Jump wires
- ❖ Ultrasonic sensor
- ❖ Breadboard

Structure of the project

The structure of the project is shown below:



Procedure

The steps to connect the Ultrasonic sensor to the board are listed below:

- ❖ Connect 5V pin of the sensor to 5V of the Arduino board.
- ❖ Connect the GND pin of the sensor to the GND of the Arduino board.
- ❖ Connect the SIG pin of the sensor to pin 8 of the Arduino board.

Sketch

Consider the below code:

1. `const int pinTOping = 7;`
2. `void setup()`
3. `{`

```

4. Serial.begin(9600); //Serial communication at 9600 bps rate
5. Serial.println("Test for the Ultrasonic range sensor"); }
6. void loop() {
7. // initializing the variables using long data type
8. long TIMEduration, in, cm;
9. // The PING))) is triggered by a HIGH pulse val2 of 2 or more microseconds.
10. // We have specified short time to clean HIGH pulse:
11. pinMode(pinTOping, OUTPUT);
12. digitalWrite(pinTOping, LOW);
13. delayMicroseconds(4);
14. digitalWrite(pinTOping, HIGH);
15. delayMicroseconds(8);
16. digitalWrite(pinTOping, LOW);
    // We have used the same pin to read the signal from the PING)), which is a HIGH
        pulse
// the time is measured in microseconds
17. pinMode(pinTOping, INPUT);
18. TIMEduration = pulseIn(pinTOping, HIGH);
19. // convert the time into a distance
20. in = microsecondsToIn(TIMEduration);
21. cm = microsecondsToCm(TIMEduration);
22. Serial.print(in);
23. Serial.print("inches, ");
24. Serial.print(cm);
25. Serial.print("centimeters");
26. Serial.println();
27. delay(200); //time delay of 200 microseconds }
28. long microsecondsToIn(long microseconds) {
29. // there are 73.746 microseconds per inch according to PING datasheet
30. // we need to divide the distance by 2
31. //It is because the ping travels forward and bounces backward
32. return microseconds / 74 / 2; }
33. long microsecondsToCm(long microseconds) {

```

- ```
34. // The speed of sound is 340 m/s
35. return microseconds / 29 / 2; }
```

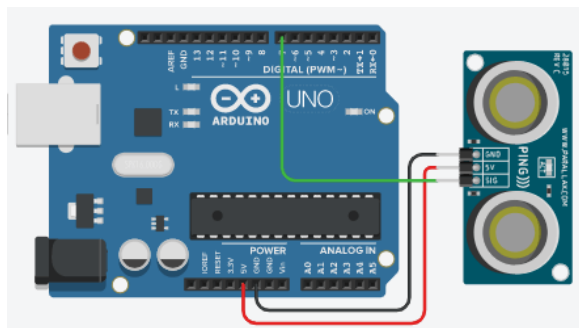
### Steps to upload the code to the project

The steps are listed below:

- ❖ Open the Arduino IDE.
- ❖ Select the type of board from Tools -> Board -> Arduino UNO.
- ❖ Select the port from Tools -> Port -> COM.
- ❖ Upload the sketch to the connection diagram.

### Connection Diagram

The connection diagram is shown below:



### Output

The output on the serial monitor will appear as:

```
Serial Monitor
Test for the Ultrasonic range sensor
43inches, 111centimeters
44inches, 113centimeters
44inches, 113centimeters
44inches, 113centimeters
44inches, 113centimeters
44inches, 113centimeters
44inches, 113centimeters
44inches, 113centimeters
```

### Project-07

## Arduino temperature sensor

The temperature sensor in Arduino converts the surrounding temperature to voltage. It further converts the voltage to Celcius, Celcius to Fahrenheit, and prints the Fahrenheit temperature on the LCD screen.

We will use a temperature sensor (TMP 36) of low voltage. Such sensors are also stable while dealing with large capacitive loads. It is also suitable for automotive applications.

The temperature sensors TMP 35, TMP 36, and TMP 37 are the sensors with the same features.

The operating voltage of the TMP 36 sensor ranges from 2.7V to 5.5V.

The sensor will look like the image shown below:



It has three terminals, which are listed below:

✓ **Pin 1: DC voltage**

Here, we will connect the DC voltage pin to 5V on the Arduino UNO board.

✓ **Pin 2: Analog voltage output**

We will consider the Analog voltage output pin as the output.

✓ **Pin 3: GND**

We will connect the GND pin to Ground on the Arduino UNO board.

*Let's start the project.*

### **Hardware Required**

The components required for the project are listed below:



- ❖ 1 x TMP 36 sensor (Temperature sensor)
- ❖ 1 x LCD display
- ❖ Arduino UNO R3 board (We can take any Arduino board).
- ❖ Jump wires

## Principle

We will connect the LCD Display and TMP 36 temperature sensor with the Arduino UNO R3 board. The sensor detects the surrounding temperature and converts it into volts, to Celsius to Fahrenheit, and displays Fahrenheit temperature on the LCD screen.

We need to open the URL: [Arduino LCD display](#) for details about LCD display.

## Connection

The steps to set up the connection are listed below:

- Connect the RS pin of LCD to pin 13 of the Arduino board.
- Connect the Enable pin of LCD to pin 12 of the Arduino board.
- Connect the D4 pin of LCD to pin 6 of the Arduino board.
- Connect the D5 pin of LCD to pin 4 of the Arduino board.
- Connect the D6 pin of LCD to pin 3 of the Arduino board.
- Connect the D7 pin of LCD to pin 2 of the Arduino board.
- Connect the Vo pin of LCD to pin 8 of the Arduino board.
- Connect the middle terminal to a sensor to A0(analog pin).
- Connect one end of the sensor to GND and another end to 5V.
- Connect one end of a resistor to the A and K of the LCD and another end to 5V.

## Sketch

Consider the below code:

```

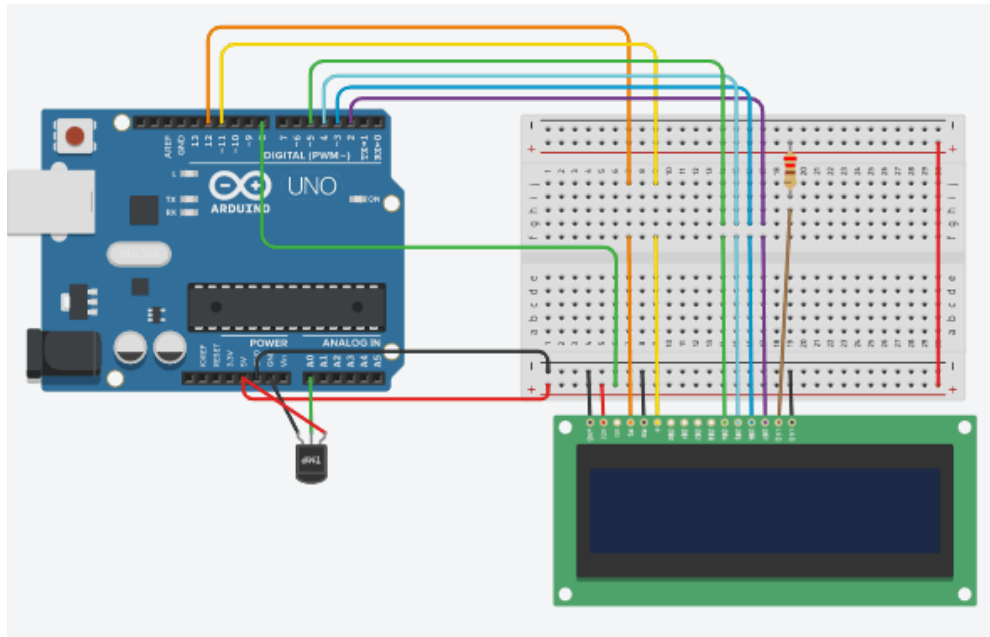
1. #include <LiquidCrystal.h>
2. // initialize the library with the pins on the Arduino board
3. LiquidCrystal lcd(13, 12, 6, 4, 3, 2);
4. const int temperature = A0; //A0 is the analog pin
5. const int D = 8; // Vo of LCD is connected to pin 8 of the Arduino
6. void setup()
7. {
8. lcd.begin(16, 2);
9. Serial.begin(9600);
10. pinMode(D, OUTPUT);
11. }
12. void loop()
13. {
14. digitalWrite(D, LOW);
15. int Temp = analogRead(temperature);
16. float volts = (Temp / 965.0) * 5;
17. float celcius = (volts - 0.5) * 100;
18. float fahrenheit = (celcius * 9 / 5 + 32);
19. Serial.println(fahrenheit);
20. lcd.setCursor(5, 0);
21. lcd.print(fahrenheit);
22. delay(2000);
23. // time delay of 2000 microseconds or 2 seconds
24. }

```

### Connection Diagram

We will show the connection using the Simulator because the connections become clearer and more precise.

We can make the same connection using the hardware devices.



## Output

The output is now visible on the LCD screen.

For better understanding, let's consider the output on the Serial Monitor.



84.69  
84.69  
84.69  
84.69

It is the temperature in Fahrenheit.

## Project-08

## Arduino Motion Sensor

We will use a *PIR motion sensor* in this project. All objects (having temperature higher than absolute zero) emit radiations from the generated heat. These radiations cannot be detected by a human eye. Hence, electronic devices such as motion sensors, etc. are used for the detection of these radiations.

### ***What is a PIR sensor?***

The Passive Infra-Red sensors or PIR sensors detect motion or movement of an object that detect infrared radiations, such as the human body. Hence, the use of sensors is very common.

The advantages of using a PIR sensor are listed below:

- ❖ Inexpensive
- ❖ Adjustable module
- ❖ Efficient
- ❖ Small in size
- ❖ Less power consumption
- ❖ It can detect motion in the dark as well as light.

The PIR sensor is shown:

The PIR sensor has three terminals, which are listed below:

- VCC
- Digital Output
- GND (Ground)



We will connect the Vcc terminal of the sensor to the 5V on the Arduino board. The PIR's sensor output can be connected to any of the digital pins on the Arduino board.

The applications of the PIR sensor are automation, security systems, etc. Such sensors work great in detecting the entrance of a person in an area and leaving it.

The detection range of PIR sensors is from 5m to 12m.

### **Working of PIR Sensors**

The working of the PIR sensor is entirely based on detecting the IR (Infra-Red) radiations, which are either emitted or reflected by the objects.

The infrared radiations are detected by the crystalline material present at the center of the sensor.

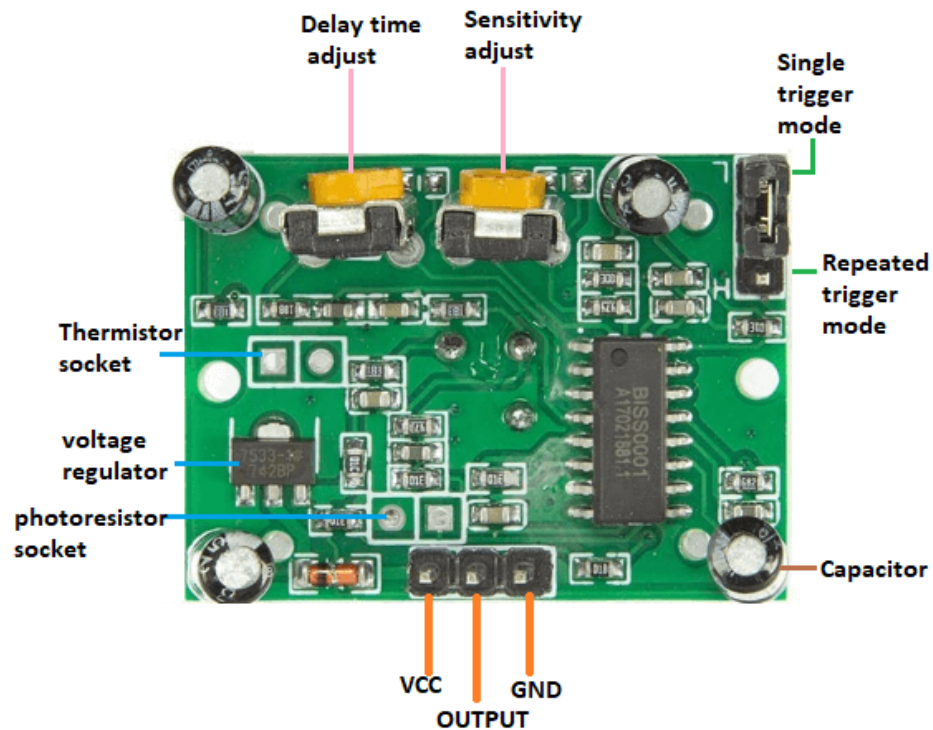
Consider a person passing in front of the background like a wall, etc. The temperature changes from room to body temperature and vice-versa within the sensor field. Changes arising in the arrival infrared radiations are converted by the sensor to the output voltage. It later detects the human body or object.

### **Structure of PIR Sensor**

A round metal can is mounted on the center with the rectangular crystal that detects the IR radiations.

A ball like a lens present on some sensors helps in enhancing the viewing angle.

The bottom part of the sensor contains many circuits mounted on it, which is shown below:



*Let's start with the project.*

## Hardware Required

The components required for the project are listed below:

- ❖ 1 x PIR motion sensor
- ❖ Arduino UNO R3 board (We can take any Arduino board).
- ❖ Jump wires
- ❖ 1 x red LED (we can take LED of any color)
- ❖ 1 x 220 Ohm resistor

## Principle

The movement of jumper present on the sensor on the L side will cause a change in the state of the sensor whenever the motion is detected. Such a condition is defined as a single trigger mode.

When the sensor resets the timer after every detection of motion, it is defined as repeated trigger mode.

The two potentiometers present on the sensor are called as Sensitivity Potentiometer and Time Potentiometer. We can adjust both the parameters (time and sensitivity) accordingly.

It should be restricted for atleast 15 seconds in front of the PIR sensor for proper calibration in the output. After 15 seconds, the sensor can easily detect movements.

If any movement is detected, the LED will be HIGH. If there is no such movement, the output will be LOW.

### **Connection**

The steps to set up the connection are listed below:

- ❖ Connect the Vcc terminal of the PIR sensor to the 5V pin of the Arduino board.
- ❖ Connect the Output terminal of the PIR sensor to pin 8 of the Arduino board.
- ❖ Connect the GND terminal of the PIR sensor to the Ground pin of the Arduino board.
- ❖ Connect the positive leg of the LED in series with 220 Ohm resistor to pin 13 of the Arduino board.
- ❖ Connect the negative terminal of the LED to the Ground pin of the Arduino board.

### **Sketch**

Consider the below code:

```
1. int LEDpin = 13; // LED pin
2. int PIRpin = 8; // The pin of Arduino connected to the PIR output
3. int PIRvalue = 0; // It specifies the status of PIR sensor
4. void setup() {
5. pinMode(LEDpin, OUTPUT);
6. pinMode(PIRpin, INPUT);
7. // the output from the sensor is considered as input for Arduino
8. Serial.begin(9600);
```

```

9. }
10. void loop()
11. {
12. PIRvalue = digitalRead(PIRpin);
13. if (PIRvalue == HIGH)
14. {
15. digitalWrite(LEDpin, HIGH);
16. // turn ON LED if the motion is detected
17. Serial.println("hello, I found you...heyyy..");
18. }
19. else
20. {
21. digitalWrite(LEDpin, LOW);
22. // LED will turn OFF if we have no motion
23. Serial.println("I cannot find you");
24. delay(1000);
25. }
26. }

```

## Steps to upload the code to the project

The steps are listed below:

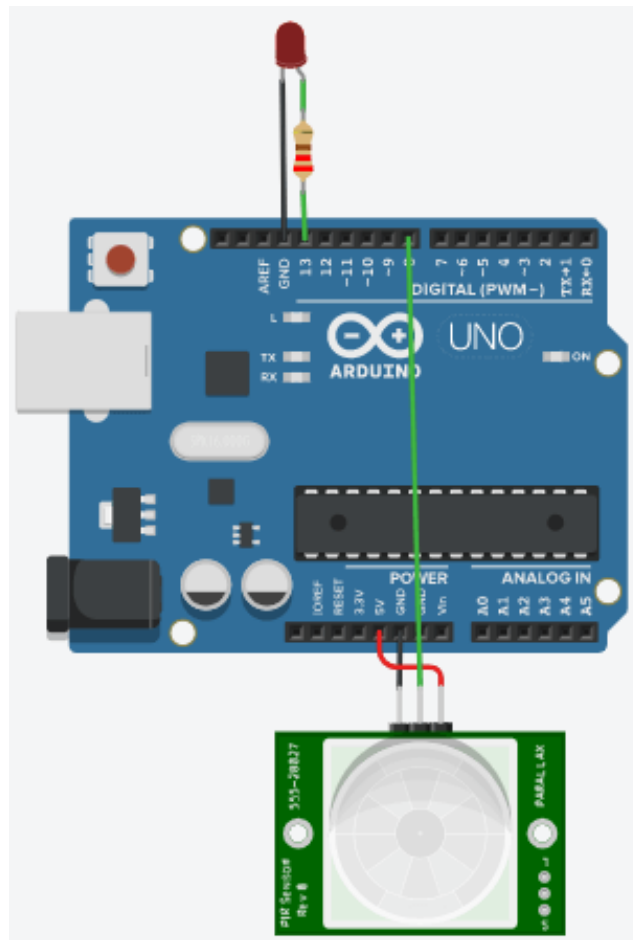
- ❖ Open the Arduino IDE.
- ❖ Select the type of board from Tools -> Board -> Arduino UNO.
- ❖ Select the port from Tools -> Port -> COM..
- ❖ Upload the sketch to the connection diagram.

## Connection Diagram

We will show the connection using the Simulator so that the connections become clearer and more precise.



We can make the same connection using the hardware devices.



The output will be based on the detection.

## **Project-09**

### **Arduino Stepper motor**

The stepper motor does not require any feedback for its operation. It can be controlled with high accuracy due to its design.

The series of magnets mounted on the shaft of the stepper motor are controlled by the electromagnetic coils. These coils are negatively and positively charged in a sequence, which makes the shaft to move in forward and backward in little steps.

We can also hold the position of the motor at any step during rotation. It has a simple, accurate open-loop system.

The Stepper is categorized into two types, which are listed below:

- ✓ Unipolar

- ✓ Bipolar

Each type possesses a different circuit, but the coding is similar.

#### ***Unipolar***

The Unipolar stepper motor consists of one winding that operates with a center tap per phase. There are three leads per phase in the motor. For the regular two-phase stepper motor, there are six leads per phase.

The Unipolar stepper motor has five leads when both the phases are joined internally.

#### ***Bipolar***

The bipolar stepper motor consists of a single winding per phase. There are two leads per phase in the motor.

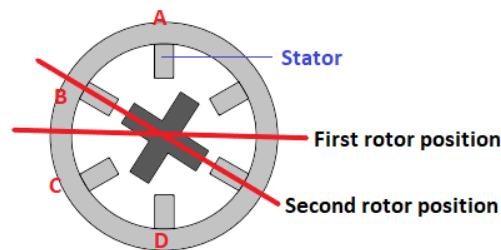
The bipolar stepper motors do not have any center tape connections. Such devices are used when we require high torque popularly at low speeds.

### ***How Stepper motor works?***

The stepper motor can control the angular position of the rotor without a closed feedback loop.

For example,

Consider a motor with six stator teeth and a rotor. It is shown below:

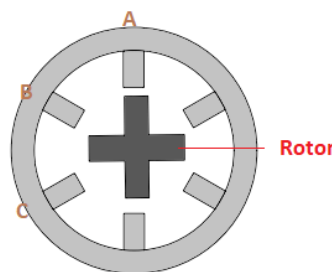


A stepper motor with six stator teeth can be triggered with three different DC power sources. The rotor in the stepper motor is made up of a stack of steel laminations. It has different teeth compared to the stator, which is four.

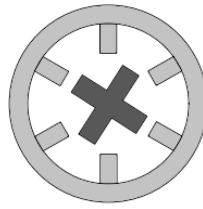
It is done so that one pair of rotor teeth at a time can be aligned easily with the stator.

If we trigger or energize the coil A and B, the rotor would rotate. The above figure signifies the step size is 30 degrees. We will energize coil B and C. After that, the coil A will energize again. It means that the rotor moves to the position with the least reluctance.

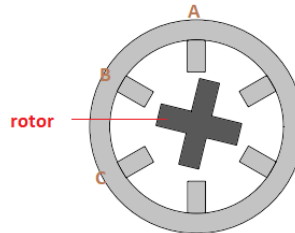
The position of the rotor, when coil A is energized is shown below:



The position of the rotor (moves 30 degrees), when coil B is energized is shown below:



When both the coils are excited, the position of the rotor (in between) is shown below:



The energizing of both the coils change the accuracy of the rotor from 30 degrees to 15 degrees.

The common stepper motor type is the hybrid motor type.

In this project, we will use the hybrid motor type. It looks like the image shown below:



## Hardware Required

The required components are listed below:

- ❖ 1 x Arduino UNO R3 (We can use any Arduino board)
- ❖ 1 x Breadboard
- ❖ Jump Wires
- ❖ 1 x 10K Ohm Potentiometer

- ❖ 1 x Stepper motor
- ❖ 1 x power supply (according to the stepper)
- ❖ U2004 Darlington Array (For a Unipolar stepper)
- ❖ SN754410ne H-Bridge (for a bipolar stepper)

## Procedure

The steps to establish the above connection are listed below:

- ❖ Connect the negative and positive terminal of the battery to the GND and 5V pin of the Arduino board.
- ❖ One outer pin of the potentiometer is connected to ground (GND), and the other external pin is connected to 5V of the Arduino board.
- ❖ The middle pin of the potentiometer is connected to the analog input pin A1 of the board.
- ❖ Connect the 8 to 11 digital pins of the Arduino board to the U2004 Darlington Array, which is further connected to the motor.
- ❖ Connect other pins of the U2004 Darlington Array to the stepper motor, as shown in the connection diagram.

## Sketch

Consider the below code:

```

1. #include <Stepper.h>
2. //library declared for the operation of stepper motor
3. const int stepsPERrevolution = 200; // We can change it according to the required
steps per
revolution for our motor
4. // the initialization of pins 8 to 11 of stepper library
 Stepper
5. myStepper(stepsPERrevolution, 8, 9, 10, 11);
6. int CountofSTEP = 0; // number of steps the motor has taken
7. void setup()
8. {

```

```

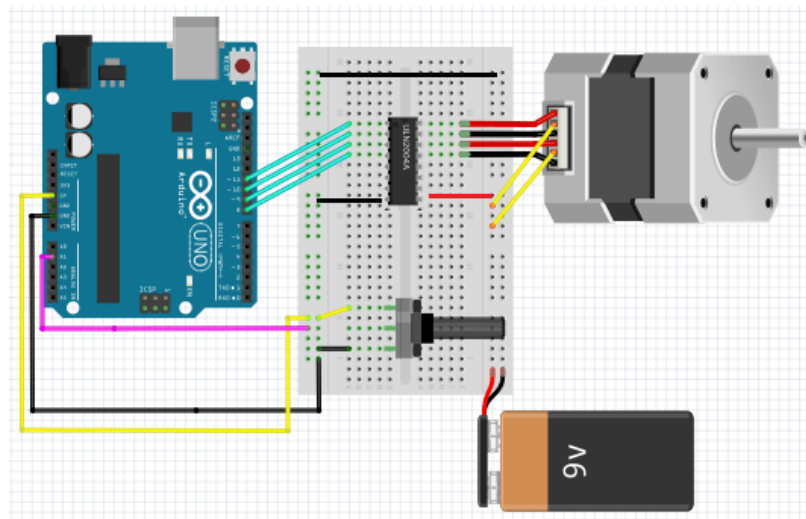
9. Serial.begin(9600);
10. }
11. void loop()
12. {
13. // read the sensor value:
14. int ReadingINSensor = analogRead(A1);
15. // we can map it to a range from 0 to 100:
16. int SpeedOFmotor = map(ReadingINSensor, 0, 1023, 0, 100);
17. // to set the speed of the motor
18. if (SpeedOFmotor > 0)
19. {
20. myStepper.setSpeed(SpeedOFmotor);
21. // step 1/100 of a revolution
22. myStepper.step(stepsPERrevolution/ 100);
23. }
24. }

```

### Connection diagram

We will show the connection using the Simulator so that the connections become clearer and more precise.

We can make the same connection using the hardware devices.



## **Project - 10**

### **Arduino Servo Motor**

The principle of the servo motor is based on *Pulse Width modulation (PWM)*. It means that the duration of pulses applied to the specific control pin controls the angle of rotation of the motor.

The construction of the servo motor is similar to a DC motor. It means that it has a rotor, stator, and control assemblies. It has closed-loop feedback for controlling the torque and speed.

The advantages of a servo motor are listed below:

- ❖ High efficiency
- ❖ High output power
- ❖ Small size
- ❖ Good power
- ❖ High precision
- ❖ rapid acceleration of loads

The applications of servo motors are machinery, automated manufacturing, robotics, radio controller airplanes, etc. The controller is considered as an essential part of the servo motor.

The movement in a servo motor is determined by an electric signal that can be either digital or analog.

The Servo library is the library that permits the Arduino to work with servo motors.

#### ***What is the Servo library, and why is it used?***

The servo library allows controlling the integrated shaft and gears. We can also position shaft at different angles between 0 and 180 degrees. The servo library on Arduino boards can support upto 12 motors, while on Arduino Mega board, it can support upto 48 motors.

It is because servos do not interfere with the functionality of PWM pins on the Arduino Mega board. On other Arduino boards, the servo library disables the PWM pin 9 and 10 even if the servo is connected to these pins.

The use of motors on Mega is also limited. It means we can use 12 motors on Arduino Mega. But, using 12 to 23 motors on the Mega board can disable the PWM functionality on the pin number 11 and 12.

### ***What is the difference between a regular motor and a servo motor?***

The difference between regular motor and servo motor are listed below:

- The output shaft of the servo motor can be moved to a specific velocity, position, and angle while regular motors cannot.
- The feedback from the motor is used by the servo motor control loop, which helps the motor to reach the desired velocity and position.

### ***What is the difference between a stepper motor and a servo motor?***

The difference between stepper motor and servo motor are listed below:

- ❖ The servo motor requires a control loop feedback. The control loop is used to monitor the current distance and velocity. Due to this, it is more reliable than the stepper motor.
- ❖ Servo motors usually have a low pole count while the stepper motors have high.
- ❖ The speed curve of the servo motor is more flexible compared to the stepper motor.
- ❖ The stepper motor has low speed and accuracy than the servo motor.

## **Project**

*Let's start the project with Arduino.*

Here, the servo motor is simply connected to the Arduino.

## **Hardware Required**

The components required for the project are listed below:



- ✓ 1 x Mini Servo motor
- ✓ Arduino UNO R3 board (We can take any Arduino board).
- ✓ Jump wires

Mini Servo Motor: It is defined as a tiny motor that can approximately rotate upto 180 degrees. It works similar to the usual servo motor, but smaller in size.

We can also use any servo motor. The connection and procedure would be the same.

## Principle

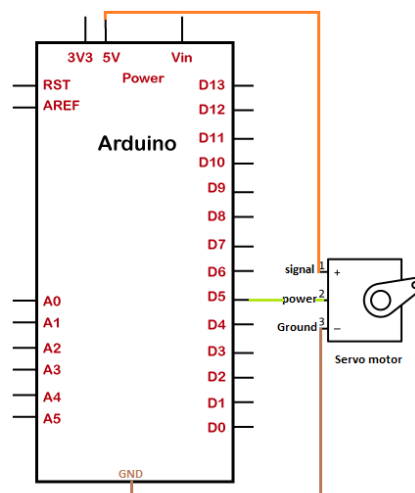
The project allows us to control the shaft at angles between 0 and 180 degrees. We can also set the rotation of the shaft at different speeds.

Servo motor has three terminals signal, power, and ground. The signal terminal is usually connected to the 5V pin of the Arduino board with the help of a wire.

The ground, power, and signal wire are represented by black, green, and red colors.

## Structure of the project

The structure of the connection or project is shown below:



## Connection

The steps to set up the connection are listed below:

- ❖ Connect the signal terminal of the servo motor to the 5V pin of the Arduino board.
- ❖ Connect the power terminal of the servo motor to pin 5 of the Arduino board. We can connect the power terminal of the motor to any digital PWM pin on the Arduino board.
- ❖ Connect the ground terminal of the servo motor to the GND pin of the Arduino board.

## Sketch

Consider the below code:

```
1. #include <Servo.h>
2. int POSservo = 0;
3. Servo servo_5;
4. void setup()
5. {
6. servo_5.attach(5); // power pin connected to pin 5 of Arduino board
7. // Here, the pin of the Arduino board connected to the servo should be a PWM pin
8. }
9. void loop() {
10. // It will sweep the servo from 0 to 180 degrees
11. for (POServo = 0; POSservo <= 180; POSservo=POServo+1) {
12. // tell servo to go to position in variable 'POServo'
13. servo_5.write(POServo);
14. // It will wait for the specified duration (20 milliseconds) to reach the position
15. delay(20); // delay of 15 millisecond(s)
16. }
17. for (POServo = 180; POSservo >= 0; POSservo= POSservo-1)
18. {
19. // It will tell servo to go to position in the declared variable 'POServo'
20. servo_5.write(POServo);
21. delay(20);
```

```
// we can modify the duration as per the requirements
22. }
23. }
```

### Steps to upload the code to the project

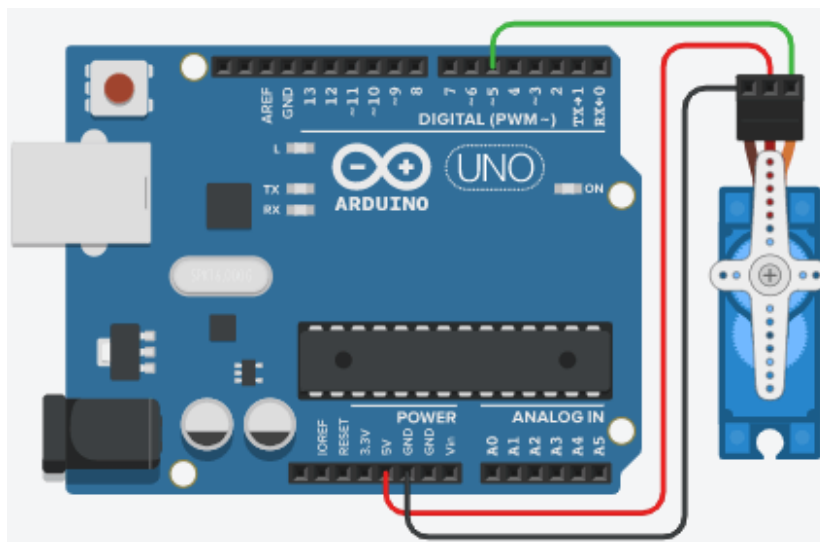
The steps are listed below:

- ❖ Open the Arduino IDE.
- ❖ Select the type of board from Tools -> Board -> Arduino UNO.
- ❖ Select the port from Tools -> Port -> COM..
- ❖ Upload the sketch to the connection diagram.

### Connection Diagram

We will show the connection using the Simulator so that the connections become clearer and more precise.

We can make the same connection using the hardware devices.



### Output

The shaft will rotate 90 degree in each direction i.e. approx. 180 degrees.

## **Project - 11**

### **Arduino DC motor**

The DC motor is considered as the simplest motor, which has various applications ranging from households to industries. Example includes an electric window in cars, electric vehicles, elevators, etc.

The principle of the DC motors is based on Electromagnetic Induction. It means that *the rotation of the motor depends on the force generated by the magnetic fields*. It converts electrical energy into mechanical energy. Such motors can be powered from the direct current.

Let's discuss how the DC motor works.

#### **Working of DC Motor**

The DC motor consists of a stator, rotor, armature, and a commutator. The commutator comes with brushes. There are two stationary magnets in the stator that are responsible for producing the magnetic field.

The armature present in the DC motor carries the alternating current. Electrical energy is converted into mechanical energy in the form of torque by the armature. It further transfers this mechanical energy via shaft.

The commutator is defined as the electrical switch. It can also reverse the direction of the current between the external circuit and the motor. The brushes act as an intermediate between the external power supply and the rotating coils.

The iron core at the center is wrapped with insulated wires concentrating on the magnetic field when current passes through the wires. The windings of insulated wire have many turns around the core of the motor.

The wire ends are connected to the commutator. The commutator further energizes the armature coils and connects the power supply and the rotating coils through brushes.

### *Advantages of DC motors*

The advantages of using DC motors are listed below:

- ❖ Low cost
- ❖ Easy motor speed control
- ❖ High reliability
- ❖ Minimal Maintenance
- ❖ High starting torque
- ❖ Quick starting
- ❖ Variable speeds
- ❖ Harmonics free

The DC motor looks like the image shown below:



*Let's start with the project.*

We will discuss two projects of the DC motor.

#### **Project - 11 (a):**

Here, we will discuss the simple connection of a DC motor with the Arduino board using diode, transistor, and resistor.

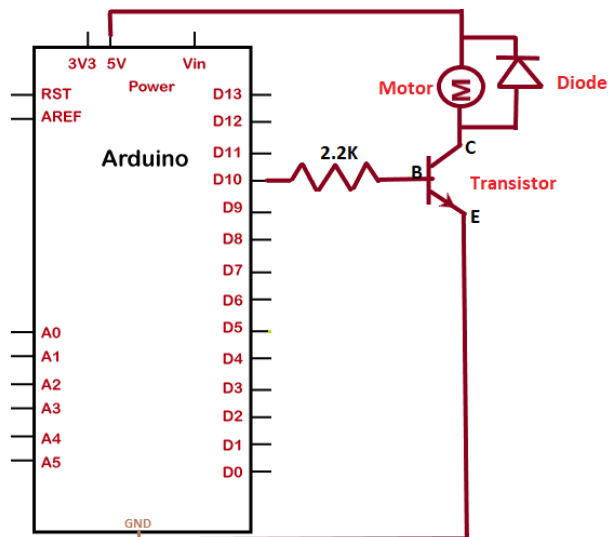
#### **Hardware Required**

The components required in the project are listed below:

- Arduino UNO R3 board
- Breadboard
- A Resistor of 2.2K Ohm
- Transistor (NPN)
- Diode
- DC Motor
- Jump wires

## Structure of the project

The structure of the project is shown below:



## Sketch

Consider the below code:

1. `int PinOFmotor = 10;`
2. `// PIN 10 of the Arduino is initialized to the variable`
3. `// the pin must be a PWM pin`
4. `void setup()`
5. `{`
6. `pinMode(PinOFmotor, OUTPUT);`

```
7. }
8. void loop()
9. {
10. digitalWrite(PinOFmotor, HIGH);
11. delay(1000);
12. digitalWrite(PinOFmotor, LOW);
13. delay(1000);
14. }
```

### **Steps to upload the code to the board**

The steps are listed below:

- ✓ Open the Arduino IDE.
- ✓ Select the type of board from Tools -> Board -> Arduino UNO.
- ✓ Select the port from Tools -> Port -> COM.
- ✓ Upload the above sketch to the connection diagram.

### **Connection**

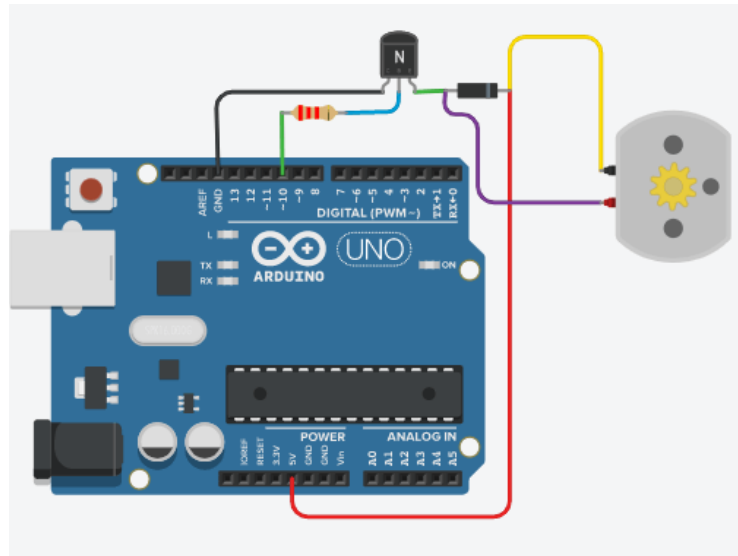
The steps to set up the connection are listed below:

- ✓ Connect one end of the resistor to pin 10 (PWM) of the Arduino board.
- ✓ Connect the other end of the resistor to the middle pin of the transistor.
- ✓ Connect one end terminal of the transistor to the GND pin of the Arduino and another end terminal to the diode.
- ✓ Connect the band facing terminal of the diode to the 5V pin of the Arduino board.
- ✓ Connect one end terminal of the DC motor to band facing terminal of the diode.
- ✓ Connect another end terminal of the DC motor to the other end of the diode.

### **Connection Diagram**

We will show the connection using the Simulator so that the connections become clearer and more precise.

We can make the same connection using the hardware devices.



After making connections, the motor will rotate.

### **Project - 11 (b):**

Here, we will discuss the connection of a DC Gear motor with the Arduino board using the L293D H-Bridge motor driver.

Let's discuss the need to use the L293D H-Bridge motor driver with the DC motor.

L293 is defined as the motor driver IC that permits the DC motor to drive in any direction. It can also simultaneously control two DC motors. It is a 16-pin Integrated Circuit (IC).

It receives signals from the microprocessor present on the Arduino board and transmits this signal to the motor. It has two VCC or voltage pins, where one pin draws current for its working and another is used to provide voltage to the DC motor.

The motor usually requires high current for its operation. We can use the microcontroller present on the Arduino, but high current might damage the microcontroller. To overcome this, the motor driver is used.

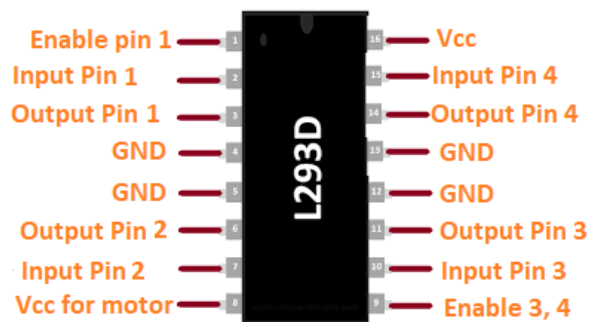


L293D is one of the most popular motor drivers used to drive the DC motors. It can run DC motors up to 1 Ampere current load.

The four outputs present on the L293D driver makes it suitable for driving the 4-wire stepper motor as well. We can also drive servo motors using the L293D driver.

### L293D Pinout

The pinout of L293D is shown below:



### Hardware Required

The components required in the project are listed below:

- Arduino UNO R3 board
- Breadboard
- DC Gear Motor
- Adjustable 30V supply
- Jump wires
- L293D H-Bridge motor driver

### Sketch

Consider the below code:

1. `#define MOTOR_1 11`
2. `// PWM pin 11`

```

3. #define MOTOR_A 10
4. // PWM pin 10
5. // we can also connect it to the other PWM pins of the Arduino
6. #define MOTOR_B 8
7. // digital I/O pin
8. #define slow 64
9. #define normal 128
10. #define fast 255
11. int Speed; // initialization of speed variable
12. // created functions
13. void Forward_Rev(void)
14. {
15. analogWrite(MOTOR_1, Speed);
16. digitalWrite(MOTOR_A, HIGH);
17. digitalWrite(MOTOR_B, LOW); }
18. void Backward_Rev(void) {
19. analogWrite(MOTOR_1, Speed);
20. digitalWrite(MOTOR_A, LOW);
21. digitalWrite(MOTOR_B, HIGH); }
22. void Forward_ramp_up(void) {
23. digitalWrite(MOTOR_A, HIGH);
24. digitalWrite(MOTOR_B, LOW);
25. for (int i=0; i<255; i++) // loop started
26. //value set from 0 to 255 {
27. analogWrite(MOTOR_A, i);
28. delay(15); // delay time in milliseconds
29. } }
30. void Forward_ramp_down(void)
31. {
32. digitalWrite(MOTOR_A, HIGH);
33. digitalWrite(MOTOR_B, LOW);
34. for (int i=255; i>=0; i--) // loop set in reverse direction
35. // value set from 255 to 0 (reverse)

```

```

36. {
37. analogWrite(MOTOR_A, i);
38. delay(15);
39. }
40. }

41. // same statement but within different function
42. void Backward_ramp_up(void) {
43. digitalWrite(MOTOR_A, LOW);
44. digitalWrite(MOTOR_B, HIGH);
45. for (int i=0; i<255; i++)
46. { analogWrite(MOTOR_A, i);
47. delay(15);
48. // we can modify the delay time as per the requirements
49. }
50. }

51. void Backward_ramp_down(void)
52. {
53. digitalWrite(MOTOR_A, LOW);
54. digitalWrite(MOTOR_B, HIGH);

55. for (int i=255; i>=0; i--)
56. {
57. analogWrite(MOTOR_A, i);
58. delay(15);
59. }
60. }

61. void Brake(void) {
62. digitalWrite(MOTOR_A, HIGH);
63. digitalWrite(MOTOR_B, HIGH); }
64. void setup() {
65. Serial.begin(9600); // bps rate of 9600
66. Serial.println("DC motor test using L293D");
67. pinMode(MOTOR_1, OUTPUT);

```

```

68. pinMode(MOTOR_A, OUTPUT);
69. pinMode(MOTOR_B, OUTPUT); }
70. void loop() {
71. Speed=slow; // Slow Speed
72. // we can modify the value as fast, slow, and normal depending on the required speed
73. //the motor will revolve according to the specified speed
74. // for example, fast will cause it to move at fast speed
75. // we can also add more parameters using #define
76. Forward_Rev();
77. delay(1000);
78. Brake();
79. delay(500);
80. Backward_Rev();
81. delay(1000);
82. Brake();
83. delay(500);
84. Forward_ramp_up();
85. Forward_ramp_down();
86. Backward_ramp_up();
87. Backward_ramp_down();
88. // the statement inside the functions will run again and again
89. // the motor will revolve in forward and backward direction
90. }

```

## Connection

The steps to set up the connection are listed below:

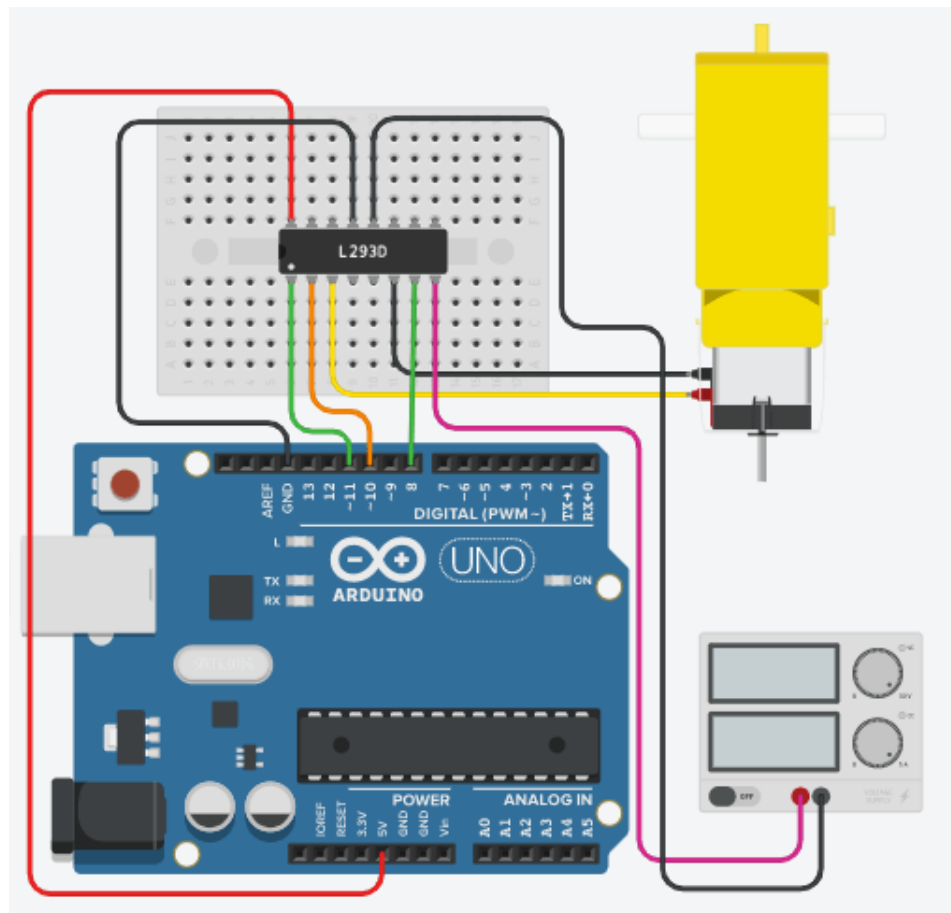
- ✓ Connect the red terminal of the power supply to the VCC of L293D.
- ✓ Connect the black terminal of the power supply to the GND of the L293D.
- ✓ Connect both terminals of the motor to Output pin 1 and 2 of the L293D driver.
- ✓ Connect input pin 2 of L293D to the digital pin 8 of the Arduino board.
- ✓ Connect the enable pin 1 and input 1 to the PWM pin 10 and 11 of the Arduino board.

- ✓ Connect VCC of the L293D driver to the 5V pin of the Arduino board.
- ✓ Connect GND of the L293D driver to the GND pin of the Arduino board.

## Connection Diagram

We will show the connection using the Simulator so that the connections become clearer and more precise.

We can make the same connection using the hardware devices.



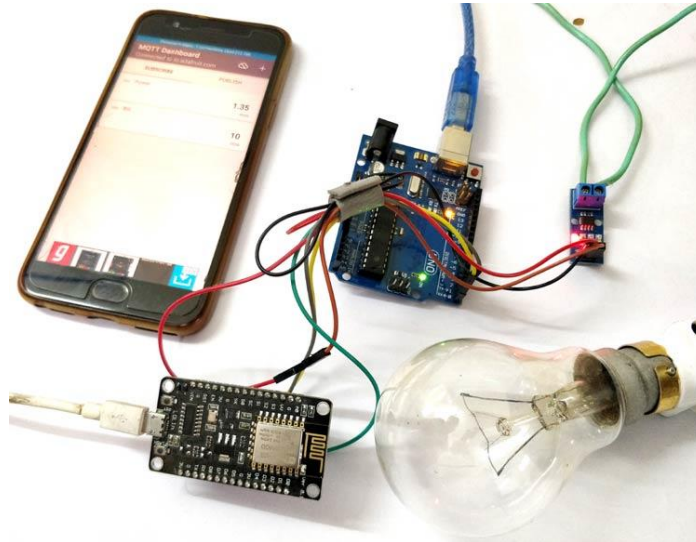
The 'Slow' mode of speed will cause the motor to rotate at a slow speed. We can also change the mode to fast or normal, as explained in the code.

We can also add more speed parameters to the code as per our requirements.

## Project – 12

### IoT Based Electricity Energy Meter using ESP12 and Arduino

---



We all know about Electricity energy meters which are installed in everyone's house or offices to measure the electricity consumption. At last of every month, many of us get worried about the high electricity bill and we have to look at the energy meter once in a while. But what if we can monitor our electricity uses from anywhere in the world and get an SMS/E-mail when your energy consumption reaches to a threshold value. Here we are building an IoT based Project of Energy Meter.

Here we have used a Current Sensor ACS712 to measure the energy consumption, we will discuss about it shortly.

We will take help of IFTTT platform to link our Wi-Fi to SMS/E-mail notifications. We will also use MQTT Dashboard Android App to monitor our Energy uses. So Lets Get started....

#### **Materials Required:**

1. Arduino Uno
2. ESP12/NodeMCU
3. ACS712-30Amp Current sensor

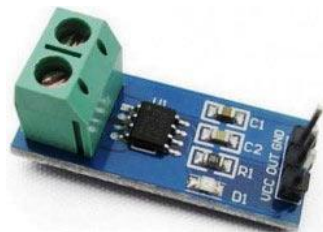
4. Any AC Appliance
5. Male-Female Wires

### **Working of ACS712 Current Sensor:**

Before we start building the project it is very important for us to understand the working of the ACS712 Current sensor as it is the key component of the project. Measuring current especially AC current is always a tough task due to the noise coupled with it improper isolation problem etc. But, with the help of this ACS712 module which was engineered by Allegro thing have become a lot easier.

This module works on the principle of Hall-effect, which was discovered by Dr. Edwin Hall. According his principle, when a current carrying conductor is placed into a magnetic field, a voltage is generated across its edges perpendicular to the directions of both the current and the magnetic field. Let us not get too deep into the concept but, simply put we use a hall sensor to measure the magnetic field around a current carrying conductor. This measurement will be in terms of millivolts which we called as the hall-voltage. This measured hall-voltage is proportional to the current that was flowing through the conductor.

The major advantage of using ACS712 Current Sensor is that is can measure both AC and DC current and it also provides isolation between the Load (AC/DC load) and Measuring Unit (Microcontroller part). As shown in the picture we have three pins on the module which are Vcc, Vout and Ground respectively.



The 2-pin terminal block is where the current carrying wire should be passed through. The module work on +5V so the Vcc should be powered by 5V and the ground should be connected to Ground of the system. The Vout pin has an offset voltage of 2500mV, meaning when there is no current flowing through the wire then the output voltage will be 2500mV and

when current flowing is positive, the voltage will be greater than 2500mV and when the current flowing is negative, the voltage will be less than 2500mV.

We will be using the Analog pin of Arduino to read the output voltage (Vout) of the module, which will be 512(2500mV) when there is no current flowing through the wire. This value will reduce as the current flows in negative direction and will increase as the current flows in positive direction. The below table will help you understand how the output voltage and ADC value varies based on the current flowing through the wire.

| Analog Value | Vout(mV)    | Current Thorough the Wires (A) |
|--------------|-------------|--------------------------------|
| 1023         | 5000        | 13.51351351                    |
| 800          | 3910.068426 | 7.621991493                    |
| 700          | 3421.309873 | 4.980053367                    |
| 512          | 2502.443793 | 0.013209691                    |
| 300          | 1466.27566  | -5.587699136                   |
| 301          | 1471.163245 | -5.561279755                   |
| 0            | 0           | -13.51351351                   |

These values were calculated based on the information given in the Datasheet of ACS712. You can also calculate them using the below formulae:

$$\text{Vout Voltage(mV)} = (\text{ADC Value} / 1023) * 5000$$

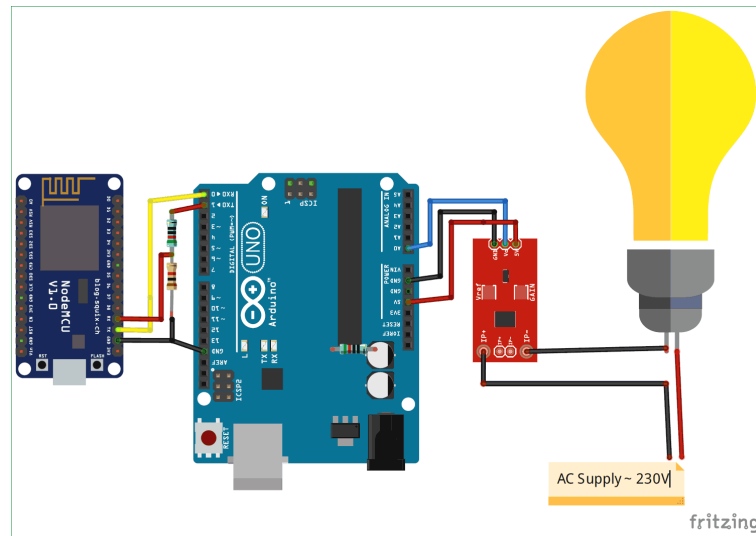
$$\text{Current Through the Wire (A)} = (\text{Vout(mv)} - 2500) / 185$$

Now, that we know how the ACS712 Sensor works and what we could expect from it. Let us proceed to the circuit diagram.

We have used this sensor to make Digital Ammeter Circuit using PIC Microcontroller and ACS712.



### Circuit diagram:



Circuit diagram for IoT based Energy Meter using Arduino and NodeMCU is given above, connect ESP12 as below:

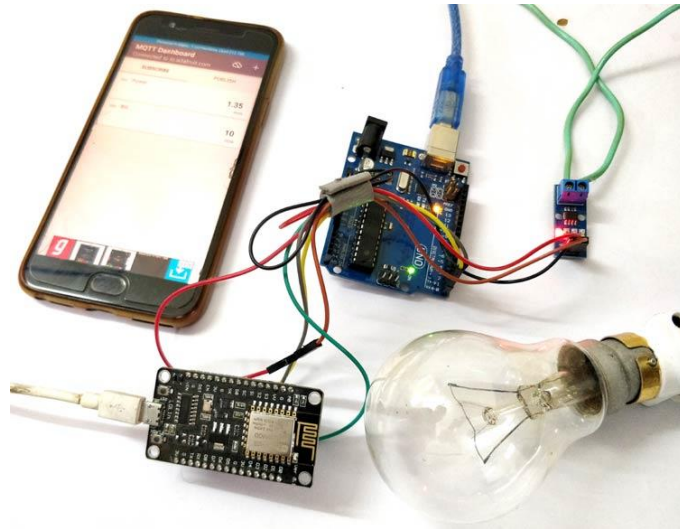
Connect Rx of ESP12 -> Tx of Arduino.

Connect Tx of ESP12 -> Rx of Arduino.

There is one analog pin available in NodeMCU (ESP12), we could use that pin but ESP series can take upto 3.3 volts on their pins. As we are using current sensor which can give upto 5 Volts so, it can damage our Wi-Fi module that's why we are not using standalone NodeMCU.

To make output of current sensor 3.3V instead of 5V, we cannot use voltage divider circuit between Current sensor and analog pin of NodeMCU because as we discussed above about the current sensor that at 2.5Volts output, current is 0Amp.

So, Arduino will read the current sensor value through analog pin and send it to the Wi-Fi module ESP12 using Serial communication. Use voltage divider circuit at receiver pin of NodeMCU so that receiver pin can get upto 3.3 Voltage level.



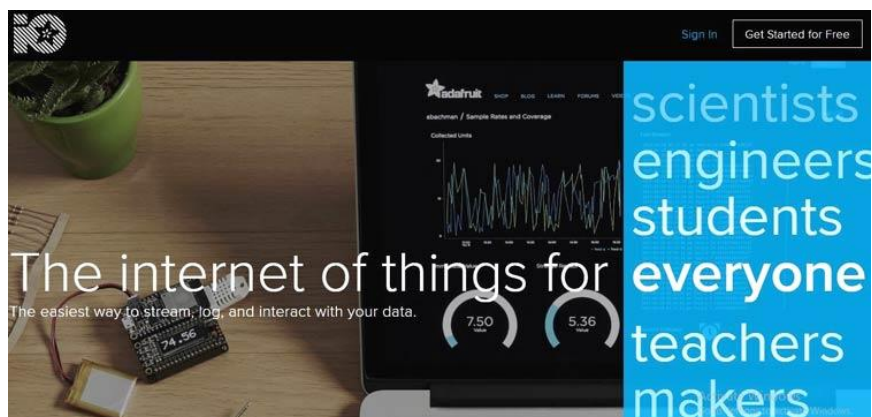
To monitor our energy uses over the internet, we have to use MQTT broker. We will use MQTT broker as AdaFruit IO platform and follow the below process to make this IoT Energy Meter

1. Setting up an AdaFruit account for storing Electricity meter readings.
2. Create Applet in IFTTT for Triggering SMS/Email for Energy Meter
3. Codes for Arduino and ESP12 Wi-Fi module.

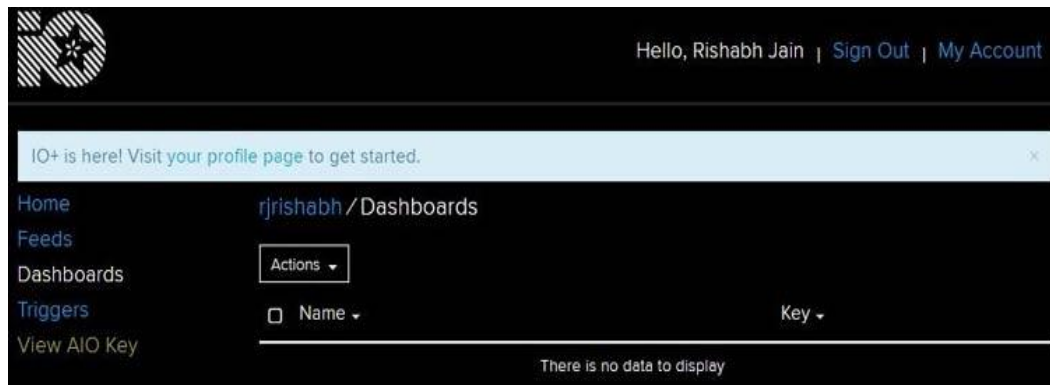
Setting up an AdaFruit account for communication:

First, we will make a feed in AdaFruit IO. Feed stores the data sent by IFTTT. To make feed follow these steps:

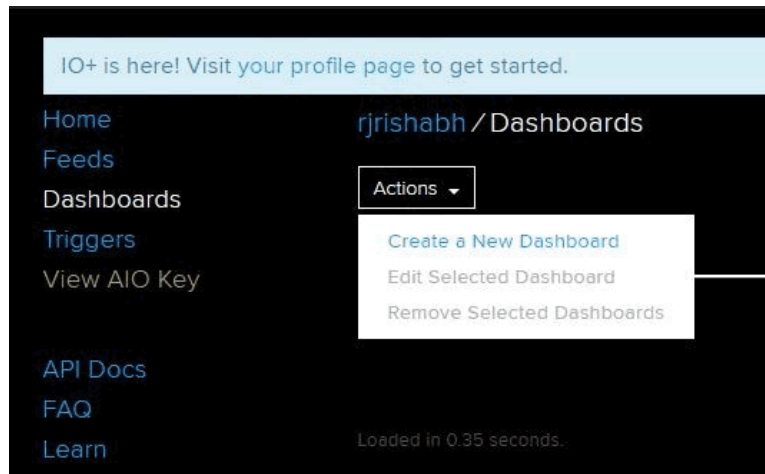
**Step 1:-** Login to Adafruit IO with your credentials or Sign up if you don't have an account.



**Step 2:-**Click on My account -> Dashboard



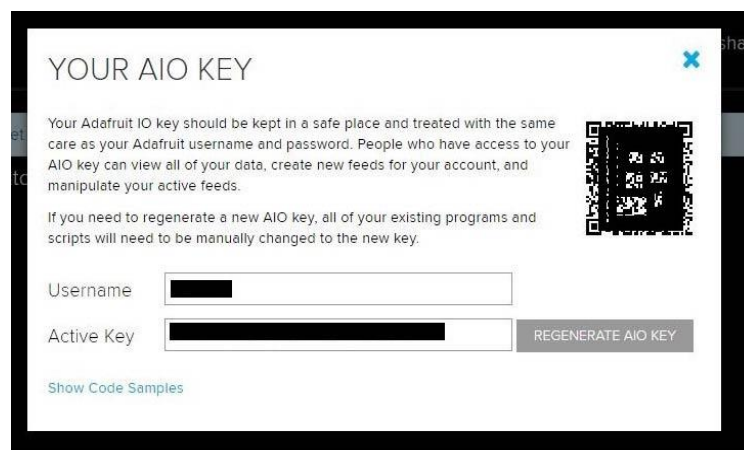
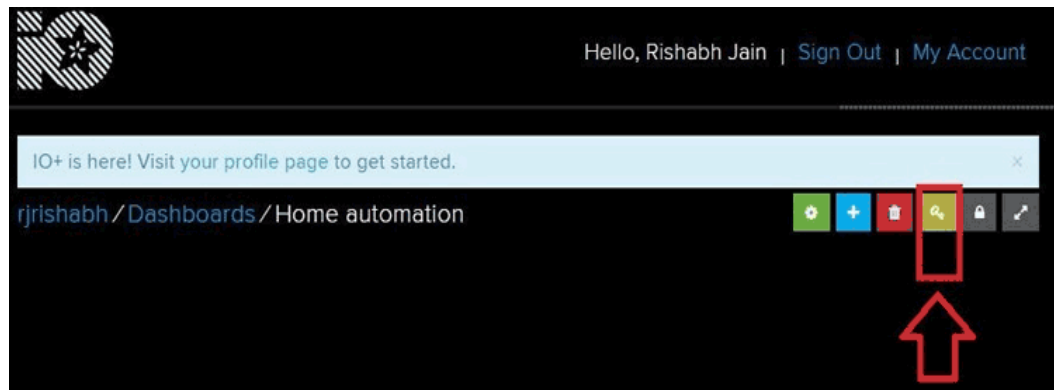
**Step 3:-**Click on Actions and Create a New Dashboard.



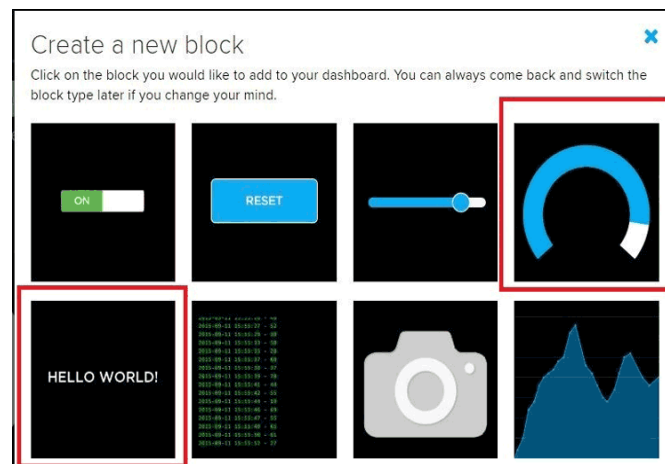
**Step 4:** Give name and description to your feed and click on Create.

A screenshot of the "Create a new Dashboard" form. The form has a title "Create a new Dashboard" and a close button. It contains two input fields: "Name" with the value "Energy\_meter1" and "Description" with the value "Monitor Energy". At the bottom right, there are two buttons: "Cancel" and "Create".

**Step 5:** Click on Key button and note down the AIO Keys, we will use this key in our code.



**Step 6:** Click on '+' button to create a new block and click on Gauge to display Energy uses level. You can also use simple text box to display energy.



**Step 7:** Now, Enter Name of Feed and click on Create. Then Select the feed and click on Next step.

### Choose feed

A gauge is a read only block type that shows a fixed range of values.

If you have lot of feeds, you may want to use the search field. You can also create a feed quickly below.

| Group / Feed                              | Last value | Recorded     |
|-------------------------------------------|------------|--------------|
| <input type="checkbox"/> light            | 0          | 7 days ago   |
| <input type="checkbox"/> chck             | 834.00     | 6 days ago   |
| <input checked="" type="checkbox"/> Power | 1.35       | 20 hours ago |

**Step 8:** In block settings, fill the min. and max values as 0 and 100 respectively or you can modify as you want.

### Block settings

In this final step, you can give your block a title and see a preview of how it will look. Customize the look and feel of your block with the remaining settings. When you are ready, click the "Create Block" button to send it to your dashboard.

Block Title

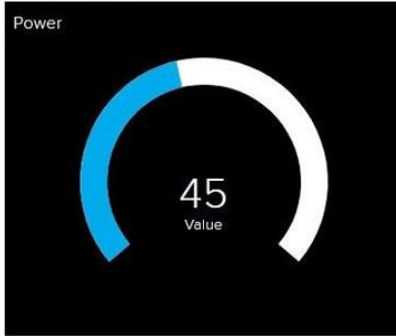
Gauge Min Value

Gauge Max Value

Gauge Width

Gauge Label

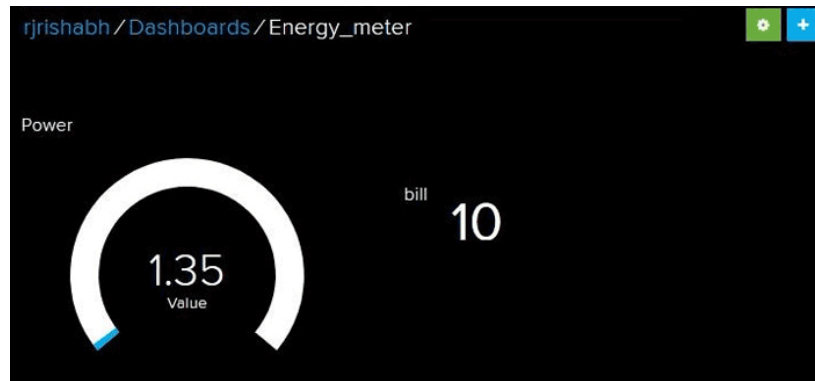
Block Preview



Previous step

Update block

**Step 9:** Your Power feed is successfully created. Now, create feed to display Bill by clicking on “+” sign.

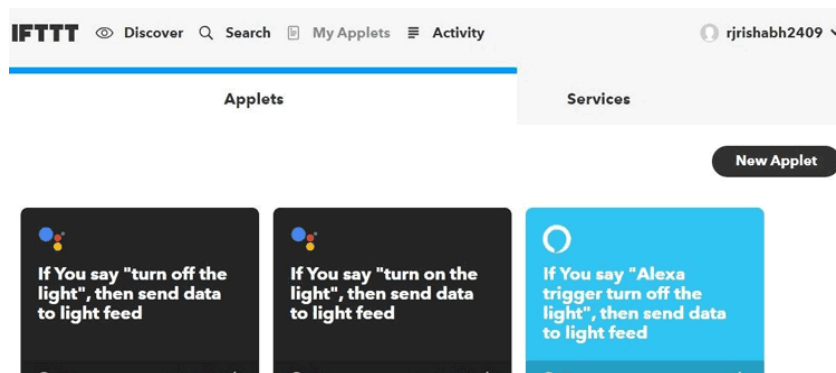


Now, we have to link AdaFruit IO to SMS/E-mail using IFTTT.

Create Applet in IFTTT for Triggering SMS/Email for Energy Meter:

**Step 1:** Login to IFTTT with your credentials.

**Step 2:** On My Applets, Click on **New Applet**




**Step 3:** Click on **+this**

**Step 4:** Search AdaFruit and click on it.

## Choose a service

Step 1 of 6



Adafruit

**Step 5:** Click on Monitor a feed on AdaFruit IO.

[< Back](#)



## Choose trigger

Step 2 of 6

### Monitor a feed on Adafruit IO

This Trigger fires anytime it validates the data that you send to your feed. Example: If Feed Temperature > 80, fire Trigger.

### Any new data

This Trigger fires any time there is new data in your feed.

**Step 6:** Choose Feed as bill, Relationship as 'equal to' and the threshold value at which you want an E-mail. Click on Create action. I have used 4 as my threshold trigger value.

### Monitor a feed on Adafruit IO

This Trigger fires anytime it validates the data that you send to your feed. Example: If Feed Temperature > 80, fire Trigger.

**Feed**

bill

The name of the feed to check.

**Relationship**

equal to

Relationship between two values.

**Value**

4

The value to compare against.

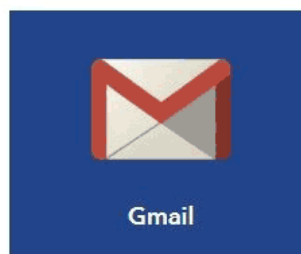
Create trigger

**Step 7:** Click on +that. Search for G-mail and click on it and Login with your g-mail credentials.

## Choose action service

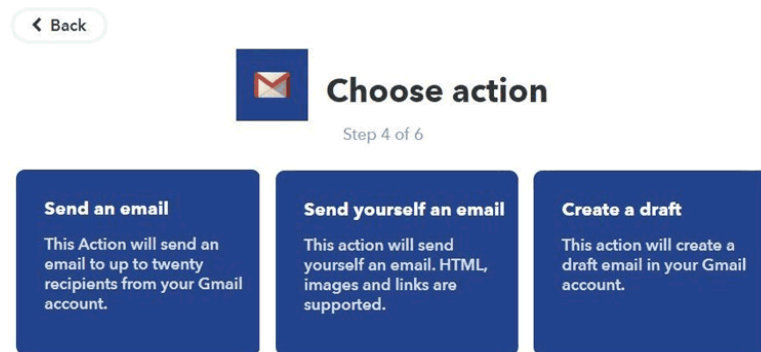
Step 3 of 6

Q G-mail

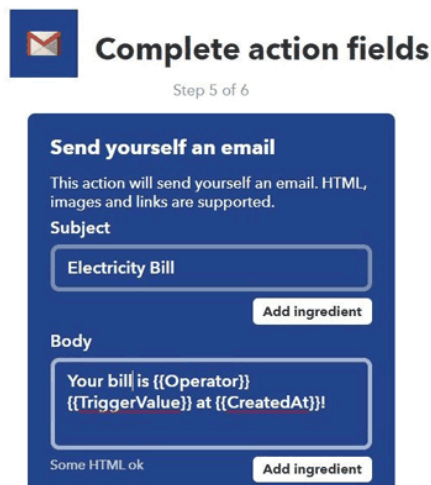




**Step 8:** Click on **send yourself an email**.



**Step 9:** Write your subject and body as shown and click to create.



**Step 10:** Your 'recipe' is ready. Review it and click on finish.



Now, we are done with web integration. Let's move on coding part..

### **Code and Explanation:**

We are using serial communication between ESP12 and Arduino. So, we have to write code for both Arduino and NodeMCU for transmitting and receiving.

### **Code for Transmitter Part i.e. for Arduino Uno:**

Complete Arduino code is given at the end of this project. We will use library for Current sensor which can be downloaded from this [Link](#).

This library has inbuilt function to calculate current. You can write your code to calculate current but this library has accurate current measuring algorithms.

First, include library for current sensor as:

```
#include "ACS712.h"
```

Make an array to store power for sending it to NodeMCU.

```
char watt[5];
```

Create an instance to use ACS712-30Amp at PIN A0. Change First argument if you are using 20Amp or 5 Amp variant.

```
ACS712 sensor(ACS712_30A, A0);
```

In setup function, define baud rate of 115200 to communicate with NodeMCU. Call sensor.calibrate() function for calibrating current sensor to get accurate readings.

```
void setup() {

 Serial.begin(115200);
```

```
sensor.calibrate();

}
```

In loop function, we will call `sensor.getCurrentAC()`; function to get the current value and store in the float variable **I**. After getting current, calculate power using  $P=V*I$  formula. We use 230V because it is the common standard in European countries, Change to your local, if necessary

```
void loop() {

 float V= 230;

 float I = sensor.getCurrentAC();

 float P = V * I;
```

These lines convert power into Wh.

```
 last_time = current_time;

 current_time = millis();

 Wh = Wh+ P *((current_time -last_time) /3600000.0) ;
```

Now, we have to convert this Wh into character form to send it to NodeMCU, for this `dtostrf()`; will convert a float to a char array so it can then be printed easily:

```
 dtostrf(Wh, 4, 2, watt);
```

The format is:

```
dtostrf(floatvar, StringLengthIncDecimalPoint, numVarsAfterDecimal, charbuf);
```

Write this character array to serial buffer using `Serial.write()`; function. This will send Wh value to NodeMCU.

```

Serial.write(watt);

delay(10000);

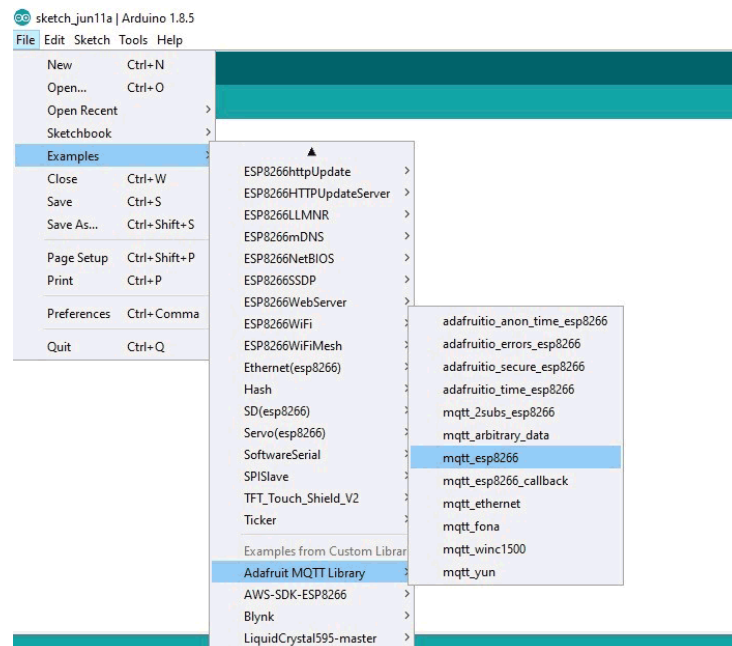
}

```

### Code for Receiver Part NodeMCU ESP12:

For this we need AdaFruit MQTT library which can be downloaded from this link.

Now, open Arduino IDE. Go to examples -> AdaFruit MQTT library -> mqtt\_esp8266



We will edit this code according to our AIO keys and Wi-Fi credentials and incoming serial data from the Arduino.

First, we included all the libraries for ESP12 Wi-Fi Module and AdaFruit MQTT.

```

#include <ESP8266WiFi.h>

#include "Adafruit_MQTT.h"

#include "Adafruit_MQTT_Client.h"

```

We define the SSID and Password for your Wi-Fi, from which you want to connect your ESP-12e.

```
#define WLAN_SSID "xxxxxxx"

#define WLAN_PASS "xxxxxxxxxxx"
```

This section defines the AdaFruit server and server port which is fixed as “io.adafruit.com” and “1883” respectively.

```
#define AIO_SERVER "io.adafruit.com"

#define AIO_SERVERPORT 1883
```

Replace these fields with your username and AIO keys which you have copied from AdaFruit site while making the Feed.

```
#define AIO_USERNAME "*****"

#define AIO_KEY "*****"
```

Then we have created an ESP12 WiFiClient class to connect to the MQTT server.

```
WiFiClient client;
```

Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.

```
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USER
NAME, AIO_KEY);
```

Setup a feed called 'Power' and 'bill' for publishing to changes.

```
Adafruit_MQTT_Publish Power = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/Power");

Adafruit_MQTT_Publish bill = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/bill");
```

In setup function, we connect Wi-Fi module to Wi-fi access point.

```
void setup() {

 Serial.begin(115200);

 delay(10);

 Serial.println(F("Adafruit MQTT demo"));

 // Connect to WiFi access point.

 Serial.println(); Serial.println();

 Serial.print("Connecting to ");

 Serial.println(WLAN_SSID);

 WiFi.begin(WLAN_SSID, WLAN_PASS);

 ...

}
```

In loop function, we will check for incoming data from the Arduino and publish this data to AdaFruit IO.

```
void loop() {

 // Ensure the connection to the MQTT server is alive (this will make the first
```

```
// connection and automatically reconnect when disconnected). See the MQTT_connect
// function definition further below.
```

```
MQTT_connect();
```

```
int i=0;
```

```
float watt1;
```

This function check for the incoming data from the Arduino and store this data into **watt[]** array using **serial.read()** function.

```
if(Serial.available() > 0){

 delay(100); //allows all serial sent to be received together

 while(Serial.available() && i<5) {

 watt[i++] = Serial.read();

 }

 watt[i++]='\0';

}
```

atof() function convert the characters to float values and we will store this float value in another float variable **watt1** .

```
watt1 = atof(watt);
```

Calculate bill amount by multiplying power (in Wh) with energy tariff and divide it by 1000 to make power in KWh.

```
bill_amount = watt1 * (energyTariff/1000); // 1unit = 1kwH
```

Now we can publish stuff!

```
Serial.print(F("\nSending Power val "));

Serial.println(watt1);

Serial.print("...");
```

This piece of code is publishing power values to the **Power** feed

```
if (! Power.publish(watt1)) {

 Serial.println(F("Failed"));

} else {

 Serial.println(F("OK!"));

}
```

This will publish electricity bill to the bill feed.

```
if (! bill.publish(bill_amount)) {

 Serial.println(F("Failed"));

} else {

 Serial.println(F("OK!"));

}
```

Our bill amount may change fast but **IFTTT takes time to trigger the applet** so these lines will give time for triggering so that we can receive threshold email.

Change the bill\_amount value on which you want to get email. Also, change in the IFTTT AdaFruit IO setup.



```
if (bill_amount==4){

 for (int i =0; i<=2; i++)

 {

 bill.publish(bill_amount);

 delay(5000);

 }

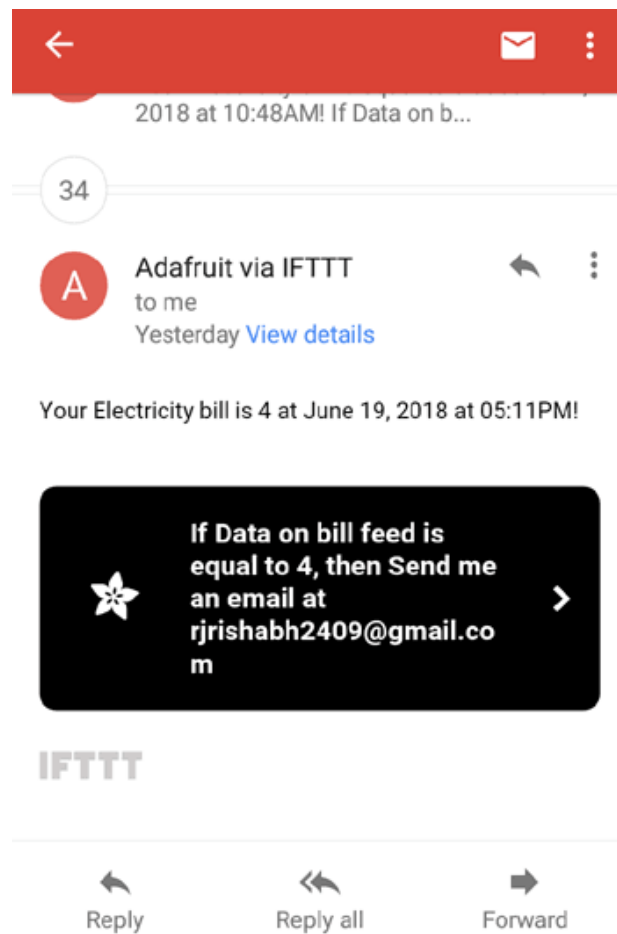
 bill_amount =6;

}
```

**Complete Code for Arduino and NodeMCU ESP12** are given at the end.

Now, upload the codes to both the boards. Connect your hardware as shown in the Circuit diagram and Open **io.adafruit.com**. Open the dashboard you just created. You will see the Power consumption and electricity Bill is updating.

When your bill reached to INR 4 then **you will get an email like this**.



### Android App for Monitoring Electricity Consumption:

You can use Android App for monitoring the values. For this download the MQTT Dashboard android app from the Play store or from this [Link](#).

To setup connection with the [io.adafruit.com](https://io.adafruit.com) follow these steps:

**Step 1:** Open the App and click on “+” sign. Fill Client Id anything you want. Server and port remain same as shown in the screenshot. You will get Username and password (Active key) from the AdaFruit IO dashboard as shown below.

The screenshot shows the MQTT Dashboard interface. On the left, a list of connections is visible, with the first one labeled '1' and 'io.adafruit.com:1883'. On the right, the 'Connection' configuration panel is open for 'Electricity Meter'. The fields are as follows:

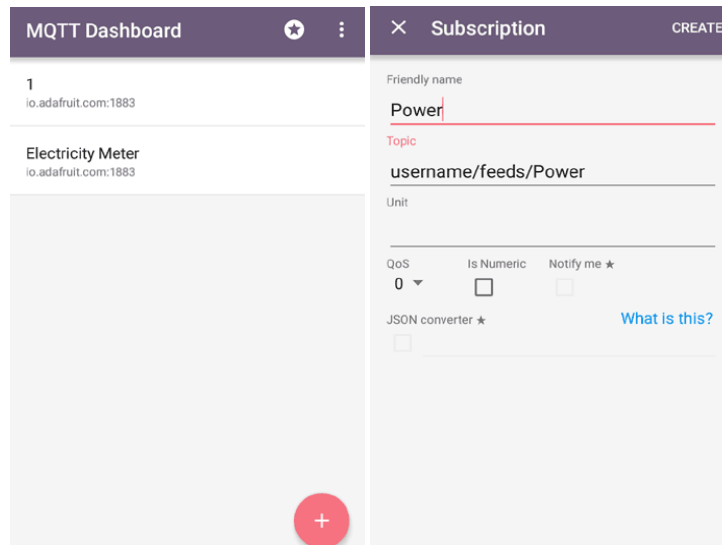
- Client ID:** Electricity Meter
- Server:** io.adafruit.com
- Port:** 1883
- Username:** [Redacted]
- Password:** [Redacted]
- SSL:** ☐
- Key store file:** Select .BKS file... (with a CLEAR button)
- Key store password:** [Redacted]

Active Key is your password.

The screenshot shows the 'YOUR AIO KEY' page. It contains the following information:

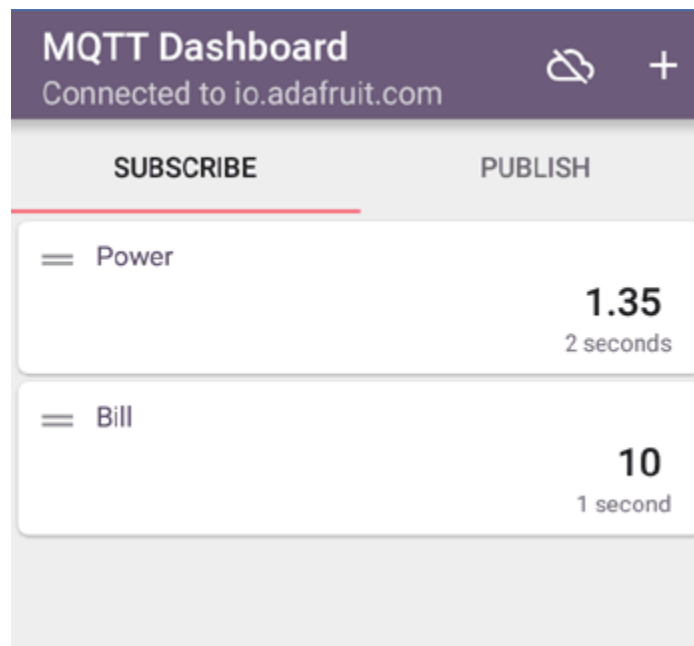
- Header:** YOUR AIO KEY
- Text:** Your Adafruit IO key should be kept in a safe place and treated with the same care as your Adafruit username and password. People who have access to your AIO key can view all of your data, create new feeds for your account, and manipulate your active feeds.
- Text:** If you need to regenerate a new AIO key, all of your existing programs and scripts will need to be manually changed to the new key.
- QR Code:** A QR code representing the AIO key.
- Form Fields:**
  - Username:** [Redacted]
  - Active Key:** [Redacted]
- Buttons:**
  - REGENERATE AIO KEY** (Grey button)
  - Show Code Samples** (Link)

**Step 2:** Select Electricity Meter and select **Subscribe**. In subscription, give friendly name and topic. Topic format is 'yourusername'/feeds/'feedname' and click on create.



**Step 3:** In the same way, make subscription for **bill** feed.

**Step 4:** As your appliances consuming energy, updated values will be displayed under the Power and Bill.



This is how you can create a Smart Electricity Energy Meter, which can be not only monitored from anywhere in the world but also trigger Email when you have high Electricity consumption.

## Code

### Code For Arduino

```
#include "ACS712.h"

char watt[5];

ACS712 sensor(ACS712_30A, A0);

unsigned long last_time =0;
unsigned long current_time =0;
float Wh =0 ;

void setup() {
 Serial.begin(115200);
 sensor.calibrate();
}

void loop() {
 float V = 230;
 float I = sensor.getCurrentAC();
 // Serial.println(I);
 float P = V * I;
 last_time = current_time;
 current_time = millis();
 Wh = Wh+ P *((current_time -last_time) /3600000.0) ;
 dtostrf(Wh, 4, 2, watt);
 Serial.write(watt);
 delay(10000);
}
```

## Code For NodeMCU

```
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

#define WLAN_SSID "a*****"
#define WLAN_PASS "*****"

char watt[5];

#define AIO_SERVER "io.adafruit.com"
#define AIO_SERVERPORT 1883
#define AIO_USERNAME "rjrishabh"
#define AIO_KEY "*****"

WiFiClient client;

int bill_amount = 0;

unsigned int energyTariff = 8.0;

Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);

Adafruit_MQTT_Publish Power = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/Power");

Adafruit_MQTT_Publish bill = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/bill");

void MQTT_connect();

void setup() {
 Serial.begin(115200);
 delay(10);
 Serial.println(F("Adafruit MQTT demo"));
 // Connect to WiFi access point.
 Serial.println(); Serial.println();
 Serial.print("Connecting to ");
 Serial.println(WLAN_SSID);
 WiFi.begin(WLAN_SSID, WLAN_PASS);
 while (WiFi.status() != WL_CONNECTED) {
 delay(500);
```

```

 Serial.print("."); }
Serial.println();
Serial.println("WiFi connected");
Serial.println("IP address: "); Serial.println(WiFi.localIP()); }
void loop() {
 // Ensure the connection to the MQTT server is alive (this will make the first
 // connection and automatically reconnect when disconnected). See the MQTT_connect
 // function definition further below.
 MQTT_connect();
 int i=0;
 float watt1;
 if(Serial.available() > 0){
 delay(100); //allows all serial sent to be received together
 while(Serial.available() && i<5) {
 watt[i++] = Serial.read();}
 watt[i++]='\0'; }
 watt1 = atof(watt);
 bill_amount = watt1 * (energyTariff/1000); // 1unit = 1kwh
 Serial.print(F("\nSending Power val "));
 Serial.println(watt1);
 Serial.print("...");
 if (! Power.publish(watt1)) {
 Serial.println(F("Failed"));
 } else {
 Serial.println(F("OK!"));
 }
 if (! bill.publish(bill_amount)) {
 Serial.println(F("Failed")); } else {
 Serial.println(F("OK!")); }
 if (bill_amount==4){
 for (int i =0; i<=2; i++){
 bill.publish(bill_amount);
 delay(5000);}

```

```

bill_amount =6;}
delay(5000);}

// Function to connect and reconnect as necessary to the MQTT server.
// Should be called in the loop function and it will take care if connecting.
void MQTT_connect() {
 int8_t ret;
 // Stop if already connected.
 if (mqtt.connected()) {
 return;}
 Serial.print("Connecting to MQTT... ");
 uint8_t retries = 3;
 while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
 Serial.println(mqtt.connectErrorString(ret));
 Serial.println("Retrying MQTT connection in 5 seconds...");
 mqtt.disconnect();
 delay(5000); // wait 5 seconds
 retries--;
 if (retries == 0) {
 // basically die and wait for WDT to reset me
 while (1); }}
 Serial.println("MQTT Connected!");}

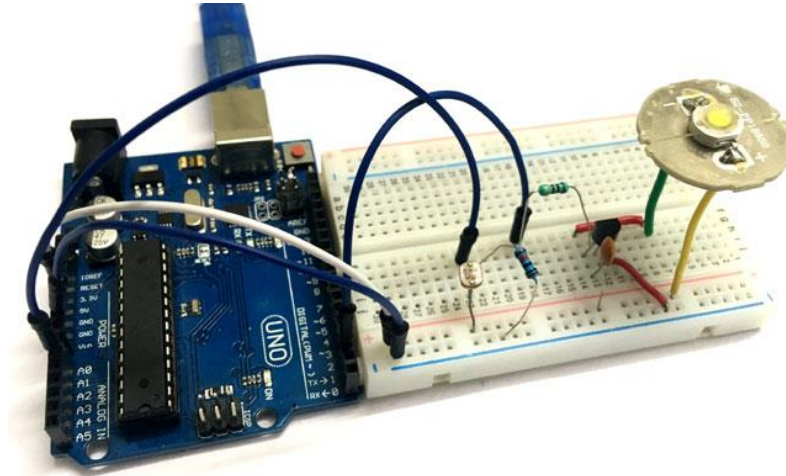
```



## Project – 13

### Auto Intensity Control of Power LED using Arduino

---



“Be a bright spark, lights off till it’s dark!” sometimes we forget to turn off the lights and waste electricity and you must have also seen street light turned on in the day. We have already built few circuits on Dark detector where lights turn off automatically if it is bright outside and turns ON if it is dark outside. But this time, in this circuit we are not only turning On and off lights based on light conditions but also varying the intensity of light according to outside light conditions. Here we have used LDR and PWM concept with Arduino for decreasing or increasing the brightness of the 1 watt Power LED automatically.

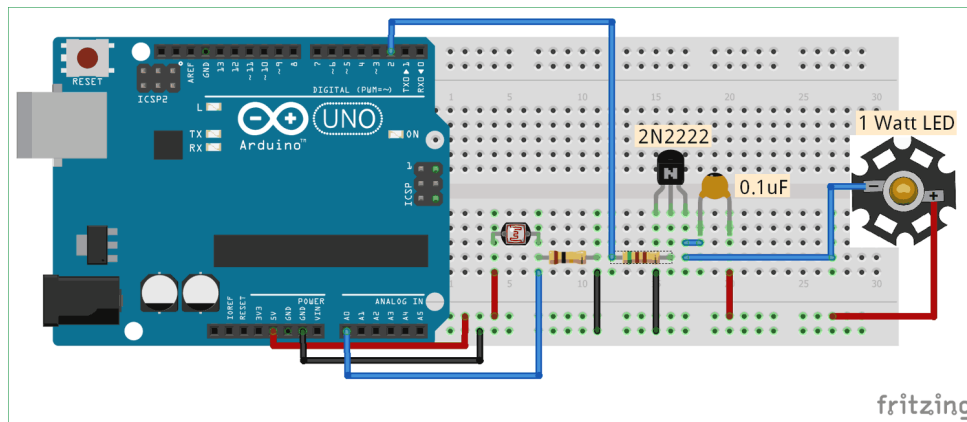
Basically, PWM refers to Pulse Width Modulation, the output signal via a PWM pin will be an analog signal and acquired as a digital signal from the Arduino. It uses the duty cycle of the digital wave to generate the sequential analog value for the signal. And, that signal is further used to control the brightness of the Power LED.

#### **Material Required:**

- ✓ Arduino UNO
- ✓ LDR
- ✓ Resistor (510, 100k ohm)
- ✓ Capacitor (0.1uF)

- ✓ Transistor 2N2222
- ✓ 1 watt Power LED
- ✓ Connecting wires
- ✓ Breadboard

### Circuit Diagram:



### Code and Explanation:

The complete Arduino code for Automatic LED dimmer is given at the end.

In the below code, we are defining the PWM pin and the variables to be used in the code.

```
int pwmPin = 2; // assigns pin 12 to variable pwm
int LDR = A0; // assigns analog input A0 to variable pot
int c1 = 0; // declares variable c1
int c2 = 0; // declares variable c2
```

Now, in the loop, we are first reading the value using the command “analogRead(LDR)” then save the analog input into a variable named “value”. By doing some mathematic calculation we are generating the PWM signal. Here, we are controlling the intensity of light using PWM only if the analog value is less than 500, and if it is more than 500 we completely turn off the lights.

```

int value = analogRead(LDR);

Serial.println(value);

c1= value;

c2= 500-c1; // subtracts c2 from 1000 ans saves the result in c1

if (value < 500)

{

digitalWrite(pwmPin, HIGH);

delayMicroseconds(c2);

digitalWrite(pwmPin, LOW);

delayMicroseconds(c1);

}

if (value > 500)

{

digitalWrite(2,LOW);

}

}

```

### **How it Controls the Light Intensity Automatically:**

As per the circuit diagram, we have made a voltage divider circuit using LDR and 100k resistor. The voltage divider output is feed to the analog pin of the Arduino. The analog Pin senses the voltage and gives some analog value to Arduino. The analog value changes according to the resistance of LDR. So, if is dark over the LDR, its resistance get increased and hence the voltage value (analog value) decreases. Hence, the analog value vary the PWM output or the duty cycle, and duty cycle is further proportional to intensity of light of power LED. So the light over the LDR will automatically control the intensity of Power LED. Below is the flow diagram

how this will work, upside arrow sign is indicating "increasing" and downside arrow sign is indicating "decreasing".

Intensity of light (on LDR) ↓ - Resistance↑ - Voltage at analog pin↓ - Duty cycle (PWM)↑ - Brightness of Power LED ↑

If its full bright outside (when analog value increases more than 500) the power LED turns off.

This is how you can control the intensity of light automatically using LDR.

### Code:

```
int pwmPin = 2; // assigns pin 12 to variable pwm
int pot = A0; // assigns analog input A0 to variable pot
int c1 = 0; // declares variable c1
int c2 = 0; // declares variable c2
void setup() // setup loop
{
 pinMode(pwmPin, OUTPUT);
 pinMode(pot, INPUT);
 Serial.begin(9600);
}
void loop()
{
 int value = analogRead(pot);
 Serial.println(value);
 c1= value;
 c2= 500-c1; // subtracts c2 from 1000 ans saves the result in c1
 if (value < 500)
 {

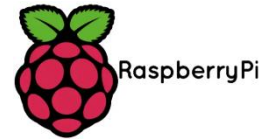
 digitalWrite(pwmPin, HIGH);
```

```
 delayMicroseconds(c2);
 digitalWrite(pwmPin, LOW);
 delayMicroseconds(c1);
}
if (value > 500)
{
 digitalWrite(2,LOW);
}
}
```





## **Raspberry Pi for Beginners**



Raspberry pi is the name of the “credit card-sized computer board” developed by the Raspberry pi foundation, based in the U.K. It gets plugged in a TV or monitor and provides a fully functional computer capability. It is aimed at imparting knowledge about computing to even younger students at the cheapest possible price. Although it is aimed at teaching computing to kids, but can be used by everyone willing to learn programming, the basics of computing, and building different projects by utilizing its versatility.

Raspberry Pi is developed by Raspberry Pi Foundation in the United Kingdom. The Raspberry Pi is a series of powerful, small single-board computers.

Raspberry Pi is launched in 2012 and there have been several iterations and variations released since then. Various versions of Raspberry Pi have been out till date. All versions consist of a Broadcom system on a chip (SoC) with an integrated ARM-compatible CPU and on-chip graphics processing unit (GPU).

The original device had a single-core Processor speed of device ranges from 700 MHz to 1.2 GHz and a memory range from 256 MB to 1 GB RAM.

To store the operating system and program memory Secure Digital (SD) cards are used. Raspbian OS which is a Linux operating system is recommended OS by Raspberry Pi Foundation. Some other third party operating systems like RISC OS Pi. Diet Pi, Kali, Linux can also be run on Raspberry Pi.

It also provides a set of general purpose input/output pins allowing you to control electronic components for physical computing and explore the Internet of Things (IOT).

Raspberry Pi, developed by Raspberry Pi Foundation in association with Broadcom, is a series of small single-board computers and perhaps the most inspiring computer available today.

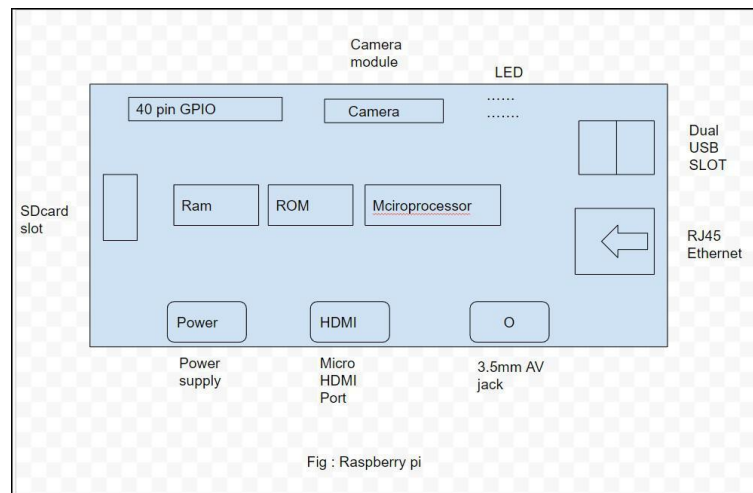
From the moment you see the shiny green circuit board of Raspberry Pi, it invites you to tinker with it, play with it, start programming, and create your own software with it. Earlier, the Raspberry Pi was used to teach basic computer science in schools but later, because of its low cost and open design, the model became far more popular than anticipated.



It is widely used to make gaming devices, fitness gadgets, weather stations, and much more. But apart from that, it is used by thousands of people of all ages who want to take their first step in computer science.

It is one of the best-selling British computers and most of the boards are made in the Sony factory in Pencoed, Wales.

### Raspberry pi Diagram :



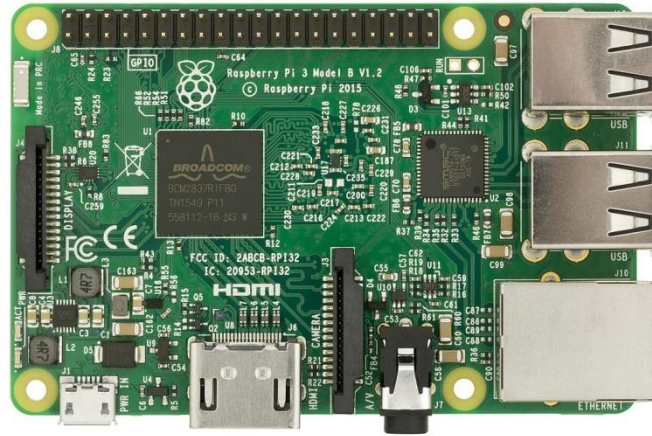
### 8.1. Raspberry Pi models:

There have been many generations of raspberry Pi from Pi 1 to Pi 4. There is generally a model A and model B. Model A is a less expensive variant and it tends to have reduce RAM and dual cores such as USB and Ethernet.

#### List of Raspberry pi models and release year:

1. pi 1 model B – 2012
2. pi 1 model A – 2013
3. pi 1 model B+ -2014
4. pi 1 model A+ – 2014
5. Pi 2 Model B – 2015
6. Pi 3 Model B- 2016
7. Pi 3 Model B+ -2018
8. Pi 3 Model A+ -2019

9. Pi 4 Model A – 2019  
10. Pi Model B – 2020  
11. Pi 400 – 2021



**Specs of the Computer:** The computer has a quad-core ARM processor that doesn't support the same instruction as an X86 desktop CPU. It has 1GB of RAM, One HDMI port, four USB ports, one Ethernet connection, Micro SD slot for storage, one combined 3.5mm audio/video port, and a Bluetooth connection. It has got a series of input and output pins that are used for making projects like – home security cameras, Encrypted Door lock, etc.

**Versatility of Raspberry Pi:** It is indeed a versatile computer and can be utilized by people from all age groups, it can be used for watching videos on YouTube, watching movies, and programming in languages like Python, Scratch, and many more. As mentioned above it has a series of I/O pins that give this board the ability to interact with its environment and hence can be utilized to build really cool and interactive projects.

## Generations and Models:

In 2012, the company launched the Raspberry Pi and the current generations of regular Raspberry Pi boards are Zero, 1, 2, 3, and 4.

Generation 1 Raspberry Pi had the following four options,

- ❖ Model A
- ❖ Model A +
- ❖ Model B
- ❖ Model B +

Among these models, the Raspberry Pi B models are the original credit-card sized format.

On the other hand, the Raspberry Pi A models have a smaller and more compact footprint and hence, these models have the reduced connectivity options.

Raspberry Pi Zero models, which come with or without GPIO (general-purpose input output) headers installed, are the most compact of all the Raspberry Pi boards types.

## 8.2. Speed Specifications:

The table below gives the speed specifications of various Raspberry Pi models and generations focusing on the version's release date, form factor and dimensions,

| Raspberry Pi Version                    | Release Date | Form Factor | Dimensions (in mm) |
|-----------------------------------------|--------------|-------------|--------------------|
| Raspberry Pi 4 Model B                  | 2019-2020    | Standard    | 85.6 x 56.5        |
| Raspberry Pi 3 Model B+                 | 2018         | Standard    | 85.6 x 56.5        |
| Raspberry Pi 3 Model B                  | 2016         | Standard    | 85.6 x 56.5        |
| Raspberry Pi 3 Model A+                 | 2018         | Compact     | 65 x 56.5          |
| Raspberry Pi Zero Wireless with Headers | 2017         | Mini        | 65 x 30 x 5        |
| Raspberry Pi Zero Wireless              | 2016         | Mini        | 65 x 30 x 5        |
| Raspberry Pi Zero                       | 2015         | Mini        | 65 x 30 x 5        |

|                          |      |          |             |
|--------------------------|------|----------|-------------|
| Raspberry Pi 2 Model B   | 2015 | Standard | 85.6 x 56.5 |
| Raspberry Pi 1 Model B + | 2014 | Standard | 85.6 x 56.5 |
| Raspberry Pi 1 Model B   | 2012 | Standard | 85.6 x 56.5 |
| Raspberry Pi 1 Model A+  | 2014 | Compact  | 65 x 56.5   |
| Raspberry Pi 1 Model A   | 2013 | Standard | 85.6 x 56.5 |

The table below gives the speed specifications of various Raspberry Pi models and generations focusing on the version's weight, General Purpose Input/Output (GPIO), central processing unit (CPU) speed, Cores and Random-access memory (RAM) –

| Raspberry Pi Version    | Weight (in grams) | GPIO   | CPU Speed | Cores | RAM            |
|-------------------------|-------------------|--------|-----------|-------|----------------|
| Raspberry Pi 4 Model B  | 46                | 40 Pin | 1.5 GHz   | Quad  | 1,2,4, or 8 GB |
| Raspberry Pi 3 Model B+ | 50                | 40 Pin | 1.4 GHz   | Quad  | 1 GB           |
| Raspberry Pi 3 Model B  | 40                | 40 Pin | 1.2 GHz   | Quad  | 1 GB           |
| Raspberry Pi 3 Model A+ | 28                | 40 Pin | 1.4 GHz   | Quad  | 512 MB         |

|                                            |    |                        |         |        |        |
|--------------------------------------------|----|------------------------|---------|--------|--------|
| Raspberry Pi Zero<br>Wireless with Headers | 10 | 40 Pin                 | 1 GHz   | Single | 512 MB |
| Raspberry Pi Zero<br>Wireless              | 10 | 40 Pin<br>Unpopulated  | 1 GHz   | Single | 512 MB |
| Raspberry Pi Zero                          | 8  | 40 Pin<br>Unpopulated  | 1 GHz   | Single | 512 MB |
| Raspberry Pi 2 Model B                     | 42 | 40 Pin                 | 1.2 GHz | Quad   | 1 GB   |
| Raspberry Pi 1 Model B +                   | 42 | 40 Pin                 | 700 MHz | Single | 512 MB |
| Raspberry Pi 1 Model B                     | 38 | 21 Pin (26 Pin Header) | 700 MHz | Single | 512 MB |
| Raspberry Pi 1 Model A+                    | 23 | 40 Pin                 | 700 MHz | Single | 512 MB |
| Raspberry Pi 1 Model A                     | 30 | 21 Pin (26 Pin Header) | 700 MHz | Single | 256 MB |

### 8.3. Connectivity Specifications:

The table below gives the connectivity specifications of various Raspberry Pi boards focusing on the version's full sized USB ports, other USB and charge methods, power and High-Definition Multimedia Interface (HDMI) ports,

| Raspberry Pi Version                    | Full sized USB Ports | Other USB & Charge Methods | Power        | HDMI Ports                |
|-----------------------------------------|----------------------|----------------------------|--------------|---------------------------|
| Raspberry Pi 4 Model B                  | 2 USB3.0 2 USB2.0    | 1 USB-C                    | 5.1V at 3A   | 2 micro-HDMI              |
| Raspberry Pi 3 Model B+                 | 4 USB2.0             | 1 MicroUSB                 | 5.1V at 2.5A | HDMI, Composite (TRRS)    |
| Raspberry Pi 3 Model B                  | 4 USB2.0             | 1 MicroUSB                 | 5.1V at 2.5A | HDMI, Composite (TRRS)    |
| Raspberry Pi 3 Model A+                 | 1 USB2.0             | 1 MicroUSB                 | 5.1V at 3A   | HDMI, Composite (TRRS)    |
| Raspberry Pi Zero Wireless with Headers | —                    | 1 MicroUSB                 | 5.1V at 1.2A | Mini-HDMI, GPIO Composite |
| Raspberry Pi Zero Wireless              | —                    | 1 MicroUSB                 | 5.1V at 1.2A | Mini-HDMI, GPIO Composite |
| Raspberry Pi Zero                       | —                    | 1 MicroUSB                 | 5.1V at 1.2A | Mini-HDMI, GPIO Composite |
| Raspberry Pi 2 Model B                  | 4 USB2.0             | 1 MicroUSB                 | 5.1V at 1.8A | HDMI, Composite (TRRS)    |

|                          |          |                    |               |                                |
|--------------------------|----------|--------------------|---------------|--------------------------------|
| Raspberry Pi 1 Model B + | 4 USB2.0 | 1 MicroUSB         | 5.1V at 1.2A  | HDMI, Composite (TRRS)         |
| Raspberry Pi 1 Model B   | 2 USB2.0 | 1 MicroUSB         | 5.1V at 3A    | PAL and NTSC, HDMI or DSI, RCA |
| Raspberry Pi 1 Model A+  | 1 USB2.0 | 1 MicroUSB or GPIO | 5.1V at 700mA | HDMI, Composite (TRRS)         |
| Raspberry Pi 1 Model A   | 1 USB2.0 | 1 MicroUSB or GPIO | 5.1V at 700mA | PAL and NTSC, HDMI or DSI, RCA |

The table below gives the connectivity specifications of various Raspberry Pi boards focusing on the version's video out quality, video in, Ethernet, bluetooth, Wi-Fi and external storage –

| Raspberry Pi Version    | Video Out Quality | Video In             | Ethernet         | Bluetooth         | Wi-Fi                      | External Storage |
|-------------------------|-------------------|----------------------|------------------|-------------------|----------------------------|------------------|
| Raspberry Pi 4 Model B  | 4kp60             | CSI Camera Connector | Gigabit Ethernet | Bluetooth 5.0     | Dual Band-2.4 GHz and 5GHz | MicroSD          |
| Raspberry Pi 3 Model B+ | 1080p60           | CSI Camera Connector | 10/100 Mbit/s    | Bluetooth 4.2/BLE | Dual Band-2.4 GHz and 5GHz | MicroSD          |

|                                         |         |                      |               |                   |                            |         |
|-----------------------------------------|---------|----------------------|---------------|-------------------|----------------------------|---------|
| Raspberry Pi 3 Model B                  | 1080p60 | CSI Camera Connector | 10/100 Mbit/s | Bluetooth 4.1     | 2.4 GHz                    | MicroSD |
| Raspberry Pi 3 Model A+                 | 1080p60 | CSI Camera Connector | —             | Bluetooth 4.2/BLE | Dual Band-2.4 GHz and 5GHz | MicroSD |
| Raspberry Pi Zero Wireless with Headers | 1080p60 | CSI Camera Connector | —             | Bluetooth 4.1     | 2.4 GHz                    | MicroSD |
| Raspberry Pi Zero Wireless              | 1080p60 | CSI Camera Connector | —             | Bluetooth 4.1     | 2.4 GHz                    | MicroSD |
| Raspberry Pi Zero                       | 1080p60 | CSI Camera Connector | —             | —                 | —                          | MicroSD |
| Raspberry Pi 2 Model B                  | 1080p60 | CSI Camera Connector | 10/100 Mbit/s | —                 | —                          | MicroSD |
| Raspberry Pi 1 Model B +                | 1080p60 | CSI Camera Connector | 10/100 Mbit/s | —                 | —                          | MicroSD |



## **8.4. History:**

Software developer Eben Upton and Software Engineers Pete Lomas and David Braden formed the Raspberry Pi foundation in 2006. The main aim of this foundation was to devise a computer to inspire children. Hence, in order to reduce the cost, the early prototypes of the Raspberry Pi were based on the 8-bit Atmel ATmega microcontroller.

On February 29th, 2012, the team started taking the orders for Model B and in the same year, they started its production run which consisted of around 10,000 units. These models were manufactured by the founders in China and Taiwan.

On February 4th, 2013, they started taking the orders for lower cost Model A. Similarly, on November 10th, 2014, the team launched for even more low-cost Model A+. The cheapest Raspberry Pi Zero was launched on November 26th, 2015.

The name Raspberry Pi was chosen with “Raspberry” as an ode to tradition of naming early computer companies after fruit. Here, "Pi" is for Python Programming Language.

## 8.5. Raspberry Pi - Getting Started:

In this modern age when computers are sleek, Raspberry Pi seems alien with tiny codes printed all over its circuit board. That's a big part of Raspberry Pi's appeal. Let us have a look at what we can do with this appealing circuit board.

### Uses:

Like a desktop computer, you can do almost anything with the Raspberry Pi. You can start and manage programs with its graphical windows desktop. It also has the shell for accepting text commands.

We can use the Raspberry Pi computer for the following –

- ❖ Playing games
- ❖ Browsing the internet
- ❖ Word processing
- ❖ Spreadsheets
- ❖ Editing photos
- ❖ Paying bills online
- ❖ Managing your accounts.

The best use of Raspberry Pi is to learn how a computer works. You can also learn how to make electronic projects or programs with it.

It comes with two programming languages, Scratch and Python. Through GPIO (general-purpose input output) pins, Raspberry Pi can be connected to other circuits, so that you can control the other devices of your choice.

### Retailers and Distributors:

Some of the global retailers from whom you can buy your Raspberry Pi computers are as follows. You can also refer to their respective websites for details about the Raspberry Pi computers.

- ✓ Electronics manufacturing company, Pimoroni ([www.Pimoroni.com](http://www.Pimoroni.com))
- ✓ Electronics store, The Pi Hut (<https://thepihut.com>)
- ✓ U.S. based electronics company, Adafruit ([www.adafruit.com](http://www.adafruit.com))

You can also get it from the following Raspberry Pi's distributors,

- ✓ Electronic components supplier, RS Components ([www.rs-components.com](http://www.rs-components.com))
- ✓ Electronic components distributor, Element14 ([www.element14.com](http://www.element14.com))

## **Requirements:**

To use your Raspberry Pi board, you need to buy a few other bits and pieces. Following is the checklist of what else we might need –

### **Monitor**

The Raspberry Pi uses a high-definition multimedia interface (HDMI) connection for video feed, and you can connect your monitor directly with this interface connection, if your monitor has an HDMI socket.

### **Television**

In the similar way, if you have High Definition Television (HD TV), you can also connect it to your Raspberry Pi using an HDMI socket. It will give you a crisper picture.

### **USB hub**

Depending on the model, Raspberry Pi has 1, 2, or 4 Universal Serial Bus (USB) sockets. You should consider using powered USB to connect other devices to your Raspberry Pi at the same time.

### **Keyboard and Mouse**

Raspberry Pi only supports the USB keyboards and mouse. If you are using keyboards and mouse with PS/2 connectors, you need to replace them with Raspberry Pi.

### **SD or MicroSD card**

As we know that the Raspberry Pi does not have a hard drive, so we need to use SD cards or MicroSD cards (depending on the model) for storage.

### **USB Wi-Fi adapter**

If you are going to use model A and A+ then, you need to buy a USB Wi-Fi adapter for connecting to the internet. This should be done because these Raspberry models do not have an Ethernet socket.

## **External hard drive**

If you want to share your collection of music and movies, you need to use an external hard drive with your Raspberry Pi model. You can connect the same by using a powered USB cable.

## **Raspberry Pi Camera Module**

The Raspberry Pi camera module originated at Raspberry Pi foundation. It is an 8MP (megapixel) fixed focus camera that can be used to shoot high-definition video and take still photos. For wildlife photography at night, it provides another version without an infrared filter.

## **Speakers**

The Raspberry Pi has a standard audio out socket. This socket is compatible with headphones and speakers that use a 3.5mm audio jack. We can plug headphones directly to it.

## **Power supply**

For power supply, it uses a Micro USB connector. Hence theoretically, it is compatible with a mobile phone and tablet charger.

## **Cables**

Following are some of the cables, which you need for the connections to the Raspberry Pi computer –

- ❖ HDMI cable
- ❖ HDMI-to-DVI adapter, if you are using a Digital Visual Interface (DVI) monitor.
- ❖ RCA cable, if you want to connect to an older television.
- ❖ Audio cable
- ❖ Ethernet cable

## **Compatible and Incompatible Devices:**

To minimize the cost, the Raspberry Pi models are designed to be used with whatever accessories we have. But, as we know that in practice, not all the devices can be compatible.

You need to check for compatible and incompatible devices as incompatible USB, keyboards and mouse can cause problems.

You can find the list of compatible and incompatible devices at [https://elinux.org/RPi\\_VerifiedPeripherals](https://elinux.org/RPi_VerifiedPeripherals)

## **Raspberry Pi - Operating System:**

Before you get started with your Raspberry Pi board, you need to provide with an OS (operating system). Linux is the most frequently used OS on the Raspberry Pi.

For using an OS, we need to create a Secure Digital (SD) or MicroSD card with an OS on it. The prerequisite for setting up the SD or MicroSD is a computer having an internet connection and the ability to **write to SD or MicroSD cards**.

### **NOOBS Software**

NOOBS means new-out-of-box software and it is the easiest way to get started with the Raspberry Pi. It is easy to copy NOOBS to your SD or MicroSD card. Once copied, it provides us with a simple menu for installing various operating systems.

There is an option to buy a card with NOOBS already installed on it, but it is always useful to know how to create your own NOOBS cards.

### **Download NOOBS**

Follow the below given steps to download NOOBS –

Step 1 – Go to the website [www.raspberrypi.org/downloads/noobs](http://www.raspberrypi.org/downloads/noobs)

Step 2 – Select from the two versions of NOOBS available. Version 1 is the main version and includes Raspbian. This is the officially supported OS, which you can use even without any network connection.

Another option is to choose the OS from the menu. You can download and install the OS from the menu, if you have a network connection. It is always recommended to download NOOBS for your first OS.

### **MicroSD card Formatting**

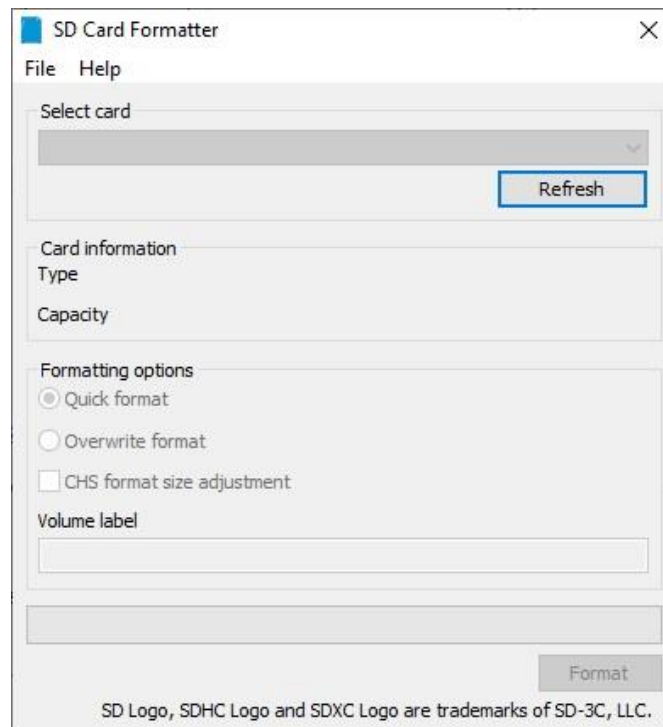
Before downloading and installing OS, we first need to format our SD or MicroSD card. We can use an application program, called SD card Formatter, from SD Association. The latest version is SD Memory Card Formatter 5.0.1.

For Windows and Mac, it can be downloaded from the link <https://www.sdcard.org/downloads/formatter/>.

Let us see how we can format the SD card by using windows, Mac OS, and Linux.

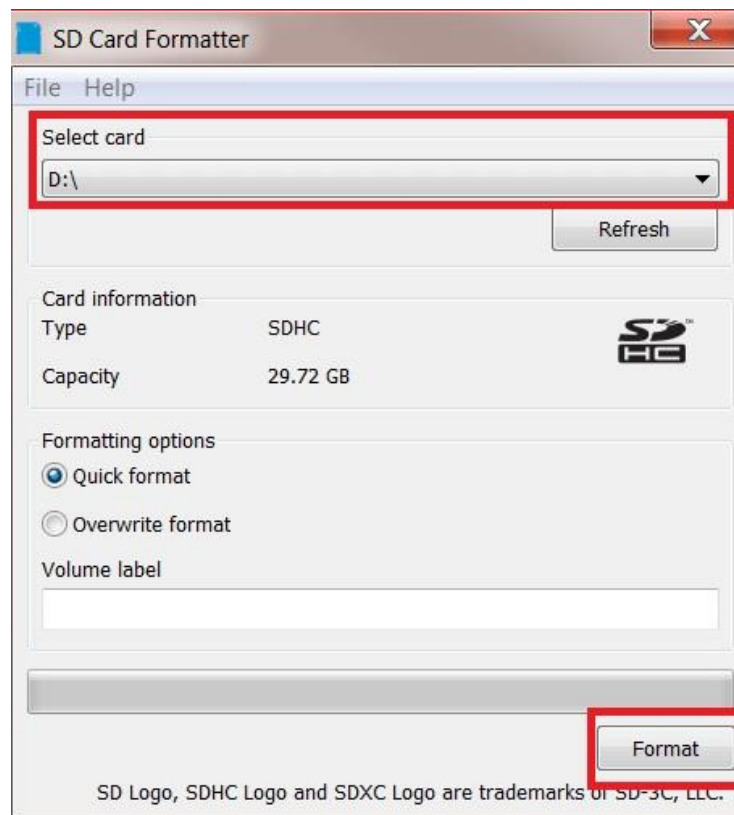
## Using Windows

**Step 1** – Download and install the SD formatter application. It will be as follows –



**Step 2** – Next, we need to select the drive in which we have our SD High Capacity SDHC/SDXC card. Once selected, click on the format button to format it.

The following screen will appear –



**Step 3** – The program will ask for the confirmation. You need to click yes to confirm the format process.

**Step 4** – Once the format process is completed, your SD card will be formatted completely.

### Using Mac OS

The process of formatting is similar as we did in windows. You just need to download and install the Mac version of SD card formatter.

### Using Linux

We will be using the GParted application program, which is an open source partition manager for Linux.

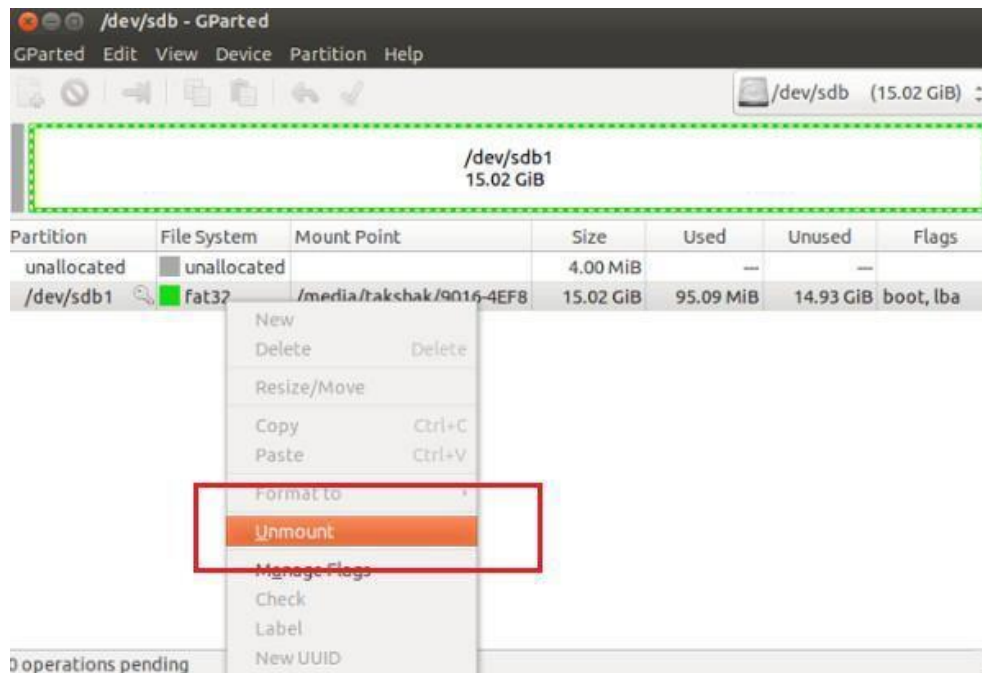
Use the steps given below to format a SD card in Ubuntu software –

**Step 1** – Download and install the GParted application by using the terminal as follows –

```
sudo apt-get install gparted
```

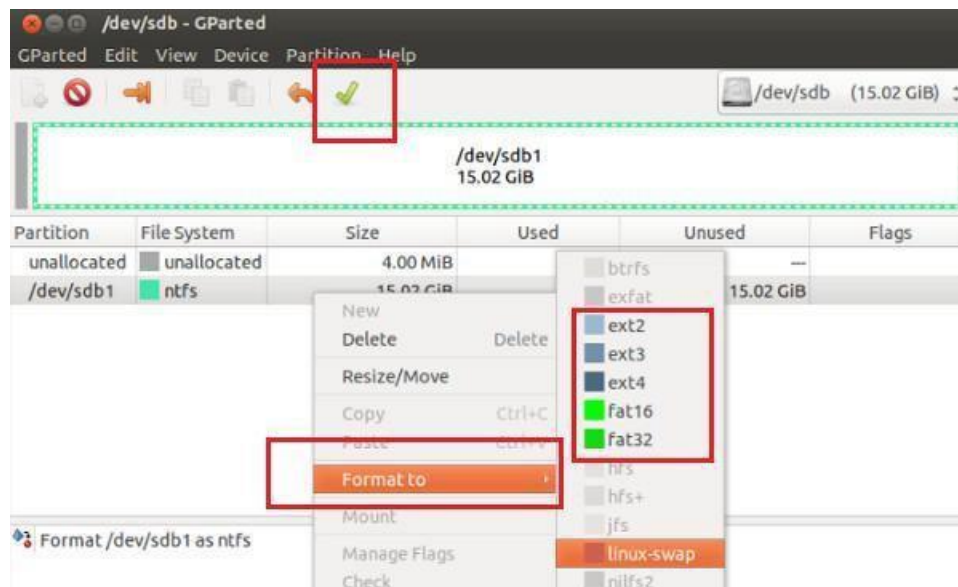
**Step 2** – Once installation is completed, you need to insert the SD card. Next, by using Unity dash, launch the GParted application.

**Step 3** – You will get the screen as below, which shows the partitions of the removable disk. But before starting the formatting, we need to unmount the disk by right-clicking on the partition as shown below –



**Step 4** – After unmounting, we need to right click on it, which will show us the Format to option. Now from the list, you can choose whatever type of file system you want on the disk.

After selecting the drive to format, you need to click on the Tick sign as shown below –



**Step 5** – It will show you a couple of warnings and the format procedure will be started.



## **Install NOOBS to Memory card**

Now, you have a formatted card and the .zip file that was downloaded from the Raspberry website. Hence, you can install NOOBS on your card.

On windows PC, you can simply double click the .zip file. It will open the file. Once opened, you can select all the files and copy them to your formatted card.

Similarly, on a Mac OS, you can see the folder that contains all the files by double clicking on the NOOBS .zip file. Now, click on the Edit menu and select all. Drag all the files onto your SD card.

In the same way, on Linux we can use the desktop environment to copy the NOOBS .zip files to our SD card.

## **Flashing a MicroSD card**

Some operating systems (OS) may not be available through NOOBS. One of them is the Reduced Instruction Set Computer (RISC) OS.

For creating a card for such an OS, we need to first download the OS as an image file. Once an image file is downloaded, we need to use the process called flashing your card. Later on, the single file can be converted into all the files which we need on our card (SD or MicroSD).

To download the OS images, we can find the links at the website <https://www.raspberrypi.org/software/>.

Now to flash the card or you can say burning an image to the card, we can use an OS image flasher Etcher. It is available for windows, Mac OS and Linux at <https://www.balena.io/etcher/>

## 8.6. Connecting Raspberry Pi:

It is quite easy to connect Raspberry Pi. Let us understand about the same in detail in this chapter.

## Ports and Sockets

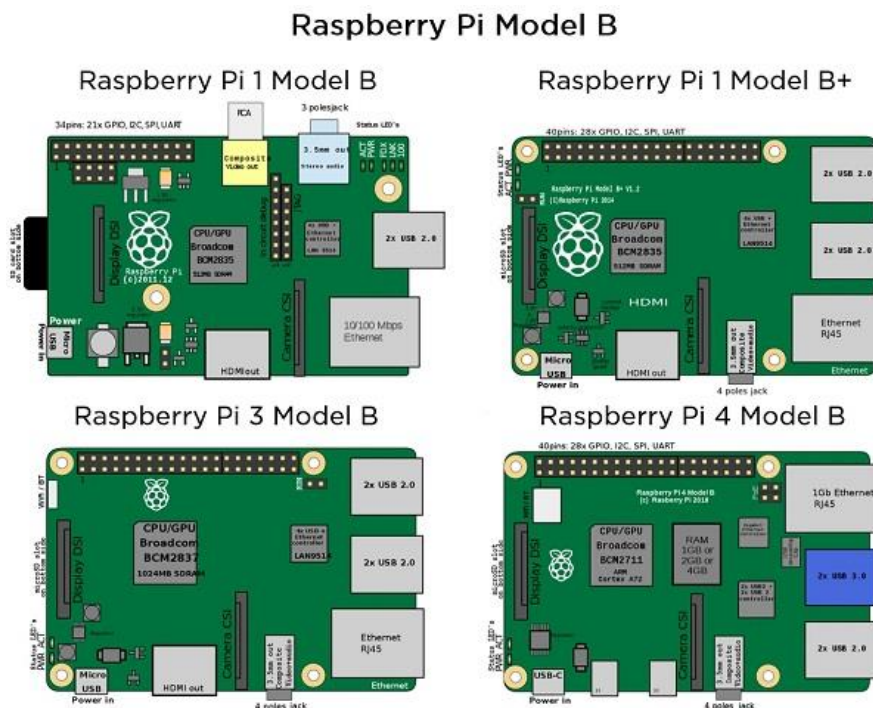
You should make sure that you have to face your Raspberry Pi in the right way. Most components and sockets, with the help of which you connect it, are sticking out at the top side whereas the back side is relatively flat. The spiky GPIO (general-purpose input output) pins should be at the top left.

Let us have a look at the diagrams below representing the location of connectors and main integrated circuits (ICs) on the Raspberry Pi boards.

The source of the diagrams is <https://core-electronics.com.au>

### Diagram 1

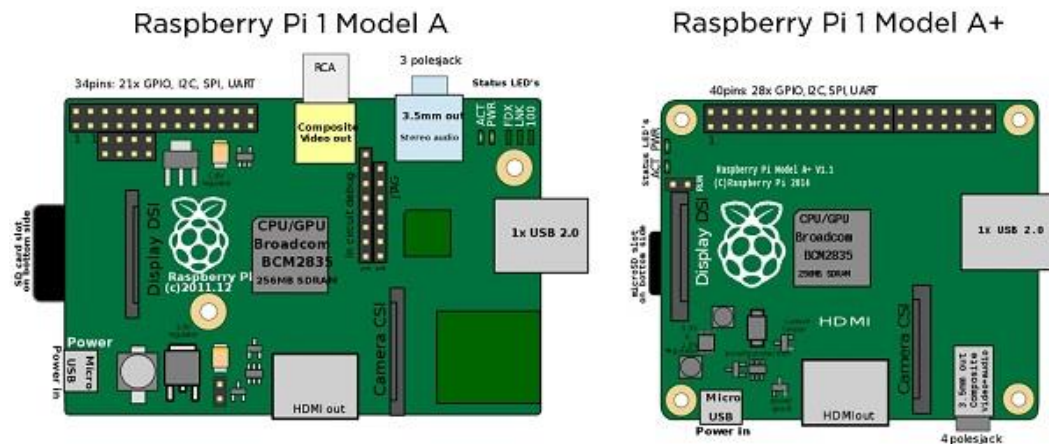
Following is the diagram for Raspberry Pi Model B –



### Diagram 2

Following is the diagram for Raspberry Pi Model A –

## Raspberry Pi Model A



**Diagram 3**

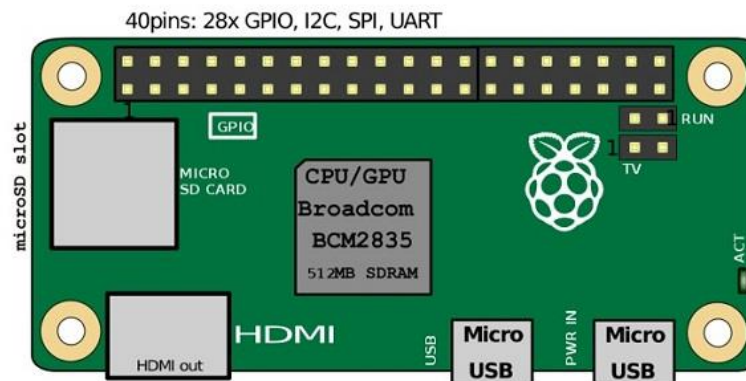
Following is the diagram for Raspberry Pi Zero –

## Raspberry Pi Zero

Raspberry Pi Zero with Headers

Raspberry Pi Zero Wireless

Raspberry Pi Zero



### Insert SD or MicroSD card

As we have discussed, you need an SD or MicroSD card with OS to get started with Raspberry Pi. We have also discussed how you can create one, in the previous chapter. Now, it is time to insert that card and get started.

If you are using model 2, 3, A+, or B+ then, you need to turn your Raspberry Pi circuit board, so that the underside is at your side and you can see that.

You can see, there would be a metal MicroSD card slot on the left side of the board. Slide your card into this slot.

On the other hand, if you are using Model A or Model B, you need an SD card and you need to flip your Raspberry Pi over. Now, slide the SD card while facing the label side above. After that you need to gently press the card home.

And we know that the models Pi Zero and Zero W have the MicroSD card slot mounted on the top surface of the board. To insert the card, you need to put the label side facing you.

## Camera Module

Camera module, an official module from the Raspberry Pi board, is a small circuit board with a strip of ribbon cable. It plugs directly into the board.



From the above diagram, you can see that for protection, the lens has a plastic film over it. You need to pull the green plastic tab to remove the film.

## On Raspberry Pi Zero

The Raspberry Pi model camera socket uses a different width of cable and you can buy that cable separately. You can also get that cable with the official Raspberry Pi Zero case. You can check the board and the camera have similar sockets for the cable.

To open the connector, you just need to gently press the connector between your finger and thumb. The camera connector is on the right of the Raspberry Pi board.

To connect the cable with the camera, insert the cable with the shiny contacts facing the camera front. And on the Pi Zero board, insert the cable with the shiny contacts facing the flat side of the board i.e., the bottom side.

## **On other Raspberry Pi Models**

To connect the camera on other boards, you need to hold the ends in between your finger and thumb. Then, gently lift the board and it will move apart to make a gap. This is the place, where you will insert the cable of the camera.

At the end of the camera's cable, you can see there are silver connectors on one side. Now hold the cable in such a way that this side faces to the left.

Once done, insert the cable into the connector on your Raspberry Pi board. Press it gently and then press the socket back together again and your board is ready with the camera.

## **Connect Raspberry Pi to Devices**

The respective processes to connect your Raspberry Pi board to different devices is explained below in detail. Let us begin by understanding how to connect a display device to your Pi board.

### **Display device**

Depending on the screen type, you have two ways to connect the display device to your Pi board. In these two ways, we are assuming that you are going to use either monitor or television. Apart from these two ways, there is an official Pi touchscreen that connects using the display socket. Let us check how we can connect an HDMI display and television, as explained below.

### **HDMI or DVI display**

The HDMI connector is on the top surface of your Raspberry Pi board. But for the Raspberry Pi Zero model, you need to use an adapter that converts Mini HDMI to an HDMI socket. For connecting, insert one end of the HDMI cable in the board or Pi ZERO connector and the other end into your monitor.

On the other hand, if you are using a DVI display, an adapter should be used.

### **Television**

If the TV you are using is having a HDMI socket, you can use that for optimal results. But if in case, your TV does not have an HDMI socket, you need to use the composite video socket.

On the Raspberry Pi Model-A and Model-B, the composite video socket is placed on the top edge of the board. It is a round, yellow-and-silver sockets.

On other models, Raspberry Pi 3, Pi 2, and Model B+, the same socket as the audio output can be used as a composite video socket. It is placed on the bottom of the board.

One thing you should note is that you will need to use a special RCA cable for this socket. Connect one end of the RCA cable to the audio output socket and the other end to Video in socket of the TV.

If you are using Pi Zero or Zero W boards then, you need to solder your own connector to the board, where it is labelled TV. This should be done because, both these boards do not have composite video socket.

### **Keyboard and Mouse**

On Raspberry Pi Model B+, Model Pi 2, and Model Pi 3 the keyboard and mouse can be directly connected. They should work fine. But for earlier models of Raspberry Pi, you should use an external USB hub to connect keyboard and mouse.

Because with this, the devices will not draw too much power from the Pi board, and we can reduce the risk of heat and other problems caused by devices.

On the other hand, for Raspberry Pi Zero, Model A, and Model A+, we must use a USB hub, since these boards have only one USB socket.

### **Audio devices**

Raspberry Pi's audio socket is a small black or blue box. On Model A and Model B, it is stuck along the top edge of the board. Whereas, on Model B+, Pi 2 and Pi 3, it is stuck along the bottom edge of the board.

If you have connected an HDMI TV, then you do not need to connect a separate audio cable, as the sound is routed through your HDMI cable.

On the other hand, if you have earphones or headphones with a 3.5mm jack, you can directly plug them into the audio socket.

Alternatively, it is recommended to use a suitable cable, as shown in the below figure. The cable has Pi's 3.5mm jack on the left and stereo input/output plugs that feed into many stereos shown on the right.



## **Internet router**

All the Raspberry Pi models other than Model A, A+, and Zero have an Ethernet socket. You can find the socket on the right edge of the Raspberry board. To connect to the internet, you can use a standard Ethernet cable in this socket.

In case if you are using a router with DHCP (Dynamic Host Configuration Protocol) support, your Raspberry Pi will automatically connect to the internet.

On the other hand, if you have a Wi-Fi adapter then, you can plug into a USB socket of Raspberry Pi and it will be ready to use whenever you turn on your board.

## **Power**

Once you are done with connecting all the necessary and required devices, it is time to connect your Raspberry Pi to power and turn it on. For this, you need to use the Micro USB power socket.

To safeguard your board from damage, you need to provide a steady 5v of power. Keep in mind that Raspberry Pi board has no on/off switch. It means, whenever you connect it with power, it will start working.

If you want to turn it off, you just need to disconnect it. So, if you want to save your data, you should proceed with caution and should shut down the Raspberry Pi first.

## **Turn on Raspberry Pi**

Connect with the power and turn on your Raspberry Pi board. There will be a rainbow of colors on screen. Afterwards, it will start to run the NOOBS software on the Memory card. You will get a choice of OS to install.

Below are the OS choices in NOOBS –

## **Raspbian**

Raspbian, a version of a Linux distribution called Debian, is the distribution that is recommended by the Raspberry Pi foundation. It has been optimized for the Raspberry Pi board.

Most of the Raspberry Pi users start with Raspbian and it includes –

- ✓ Graphical Desktop software.
- ✓ Web browser.
- ✓ Development and programming tools like Scratch, Python etc.

It has two versions, one with the PIXEL desktop and other is termed as Raspbian Lite, with a more minimal installation.

## **LibreELEC and OSMC**

Both are the versions of Kodi media center. They are mainly used for playing music and video.

## **RISC OS**

It is an alternative to Linux OS, which most of the people use on the Raspberry Pi. It has a GUI (Graphical User Interface). In 1987, it was created by Acorn Computers and now-a-days, it is maintained and managed by RISC OS open Limited.

## **Data Partition**

If you use the Data Partition option, it will give you an option to sort the data. The sorted data can be accessed by various Linux distributions.

## **Lakka**

It is a retro gaming system that includes emulators for a range of vintage home computers such as Commodore 64 and Amiga, Amstrad CPC, ZX Spectrum and various Atari machines.

It also includes emulators for a range of game consoles such as Nintendo machines and Sony PlayStation. Although the Bomberman clones and 2048 games are included but, if you want to use Lakka, you need to get the games separately.

Plug your USB with games files and you will be ready to get games into Lakka.



## **Recalbox**

It is another game system. It also includes emulators for Super Nintendo Entertainment System (SNES), Nintendo Entertainment System (NES), Game Boy Advance, PC Engine, and Sega Master System. The shareware version of a famous game called Doom is also included in the Recalbox game system.

## **Screenly OSE**

As the name implies, it is a digital signage system. It enables the users to use a Raspberry Pi with a connected HD screen as a digital sign. Here, OSE refers to Open Source Edition.

It enables the following to be displayed on the screen –

- ✓ Videos
- ✓ Images
- ✓ Web pages

Screenly OSE is also suitable for displaying the advertisements and information in public areas like shops, schools, offices, shopping malls, railway stations, etc.

## **Windows 10 IoT Core**

As the name implies, it is the version of Windows which is designed to support the IoT (Internet of things) devices. It is actually different from the windows desktop experience we are familiar with.

Once installed, it will give us the following two versions –

- ❖ RTM version – It is the release to manufacturing (RTM) version. It is recommended to use because it is a stable version as compared to the Pre-release version.
- ❖ Pre-release version – Another is pre-release version, which is less stable as compared to RTM version.

## **TLXOS**

This is ThinLinX's thin client software. It is a trial version and enables the Raspberry Pi to work as a virtual desktop. By using ThinLinX, we can also manage one or more Raspberries centrally.

## 8.7. Raspberry Pi – Configuration:

In this chapter, we will learn about configuring the Raspberry Pi. Let us begin by understanding how to configure Raspberry Pi board in Raspbian.

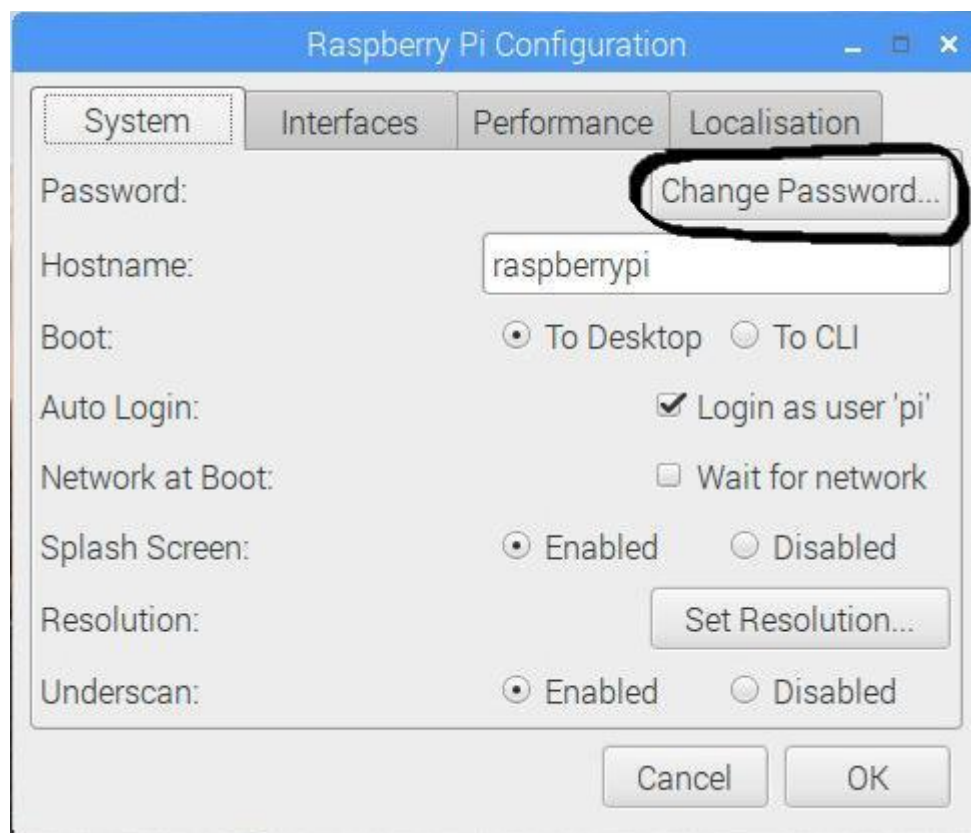
### Raspbian configuration

For configuring Raspberry Pi in Raspbian, we are using Raspbian with PIXEL desktop. It is one of the best ways to get Raspbian started with the Raspberry Pi. Once we finish booting, we will be in the PIXEL desktop environment.

Now to open the menu, you need to click the button that has the Raspberry Pi logo on it. This button will be in the top left. After clicking the button, choose Raspberry Pi configuration from the preferences.

### Configuration tool

Following is the configuration tool in PIXEL desktop –



By default, the configuration tool opens to its system tab which has the following options –

- ❖ Change Password – The default password is raspberry. You can change it by clicking the change password button.
- ❖ Change the hostname – The default name is raspberry pi. You can also change it to the name, which you want to use on the network.
- ❖ Boot – You can choose from the two options and control whether Raspberry Pi boots into the desktop or CLI i.e., command line interface.
- ❖ Auto Login – With the help of this option, you can set whether the user should automatically log in or not.
- ❖ Network at Boot – By choosing this option, you can set whether the pi user is automatically logged in or not.
- ❖ Splash screen – You can enable or disable it. On enabling, it will display the graphical splash screen that shows when Raspberry Pi is booting.
- ❖ Resolution – With the help of this option, you can configure the resolution of your screen.
- ❖ Underscan – There are two options, enable or disable. It is used to change the size of the displayed screen image to optimally fill the screen. If you see a black border around the screen, you should disable the underscan. Whereas, you should enable the underscan, if your desktop does not fit your screen.

There are three other tabs namely Interfaces, Performance, and Localization. The job of interface tab is to enable or disable various connection options on your Raspberry Pi.

You can enable the Pi camera from the interface tab. You can also set up a secure connection between computers by using SSH (short for Secure Shell) option.

If you want to remote access your Pi with a graphical interface then, you can enable RealVNC software from this tab. SPI, I2C, Serial, 1-wire, and Remote GPIO are some other interfaces you can use.

There is another tab called Performance, which will give you access to the options for overclocking and changing the GPU memory.

The localization tab, as the name implies, enable us to set –

- ❖ The character set used in our language.
- ❖ Our time zone.
- ❖ The keyboard setup as per our choice.

- ❖ Our Wi-Fi country.

## **Configure Wi-Fi**

You can check at the top right, there would be icons for Bluetooth and Wi-Fi. The fan-shaped icon is on the Wi-Fi. To configure your Wi-Fi, you need to click on that icon. Once clicked, it will open a menu showing the available networks. It also shows the option to turn off your Wi-Fi.

Among those available networks, you need to select a network. After selecting, it will prompt for entering the Wi-Fi password i.e., the Pre Shared Key.

If you see a red cross on the icon, it means your connection has been failed or dropped. To test whether your Wi-Fi is working correctly, open a web browser and visit a web page.

## **Configure Bluetooth Devices**

We can use wireless Bluetooth devices such as keyboard and/or mouse with Pi 3 and Pi zero W because these models are Bluetooth-enabled. In PIXEL desktop, you can set up your Bluetooth devices easily.

Following are the steps to configure the Bluetooth devices –

- ❖ First, make your device discoverable for pairing.
- ❖ Now, you need to click on the Bluetooth menu at the top right of the screen. It is aligned to the Wi-Fi button.
- ❖ Now, choose the Add Device option.
- ❖ The Raspberry will start searching for the devices and when it finds your device, click it and click the pair button.

## **Data Partition Setup**

As we know that data partition is that area on your memory card (SD or MicroSD) which can be shared by various distributions. One of the best examples of use of a data partition is transferring the files between distributions.

The data partition has the label data.

You can use this labeled data to make a directory point to it as follows –

**Step 1** – First, you need to boot the Raspberry Pi into Raspbian.

**Step 2** – Now, click the Terminal icon to get to the command line.

**Step 3** – Next, type the command `mkdir shared`. It will create a directory named `shared`.

**Step 4** – Write the command `sudo mount -L data shared`. This command will point the directory to the shared partition.

**Step 5** – Write the command `sudo chown $USER: shared`. It will set the permission for writing in this shared folder.

**Step 6** – Now, to go to this shared folder, you need to type the command `cd shared`.

Once all the files are created in this shared folder, they will be available to all the distributions that have the permission to access the data partition.

## 8.8. Raspberry Pi - Working with Linux:

This chapter enlightens about the functioning of Raspberry Pi with Linux.

### PIXEL Desktop Environment

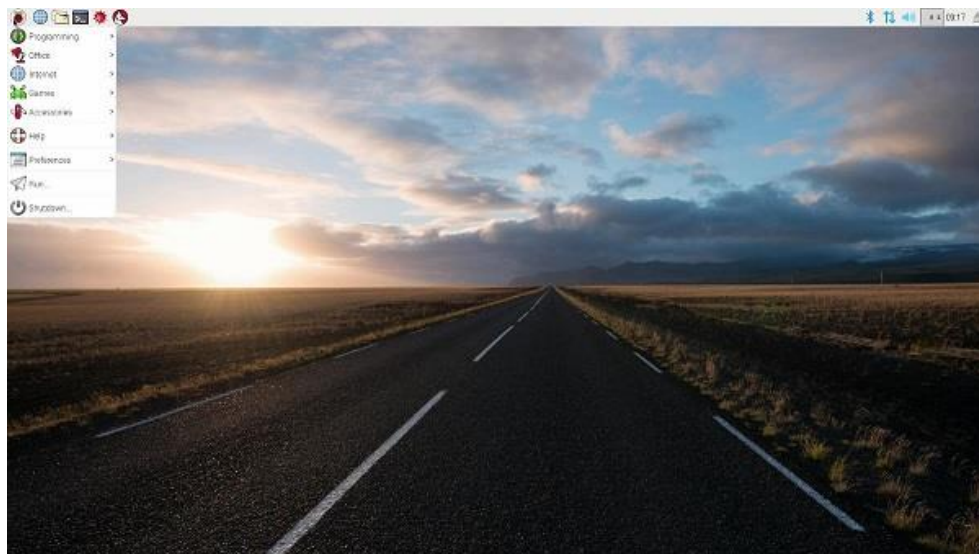
PIXEL (Pi Improved Xwindows Environment, Lightweight) is a visual desktop environment which is a part of the recommended Raspbian Linux distribution. It is the quickest way to get started with Raspberry Pi and by default, it appears when our Raspberry Pi computer finishes starting up.

Some of the characteristics of PIXEL are as follows –

- ❖ It is based on LXDE (Lightweight X11 Desktop Environment) open-source desktop.
- ❖ Raspberry Pi foundation has redesigned LXDE and converted it into a PIXEL desktop environment.
- ❖ The PIXEL desktop environment works in a similar way to Mac OS and Windows OS.
- ❖ We can manage and find files by using mouse and icons.
- ❖ Using this desktop environment, it's really intuitive to navigate.

### Navigate Desktop Environment

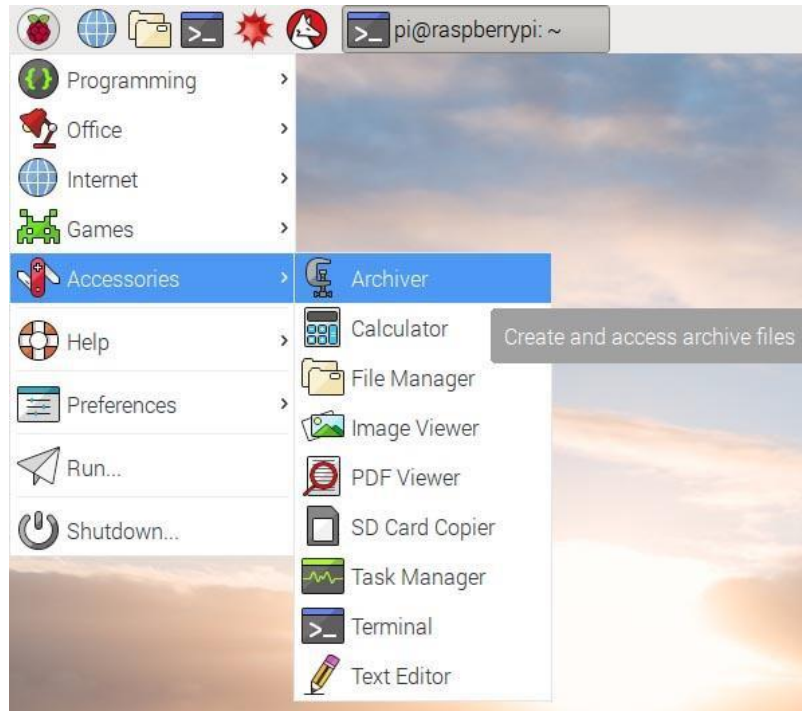
The image below is of the PIXEL desktop environment. You can see a taskbar (a strip along the top of the screen), which is usually visible in every program we will be using.



## The Application Menu

For most of the programs, which we would like to run under PIXEL desktop or any other desktop environment, we need to use the application menu. You can get it by clicking the Raspberry Pi icon at the top left side of the desktop screen.

You will see the image as follows –



## Submenu Programs

You will get the submenu program on the right, after moving the cursor over the categories of the programs. It will show the programs in that particular category.

You need to click on that category to start with that. If you want to add that category icon to the desktop, just right-click that program on the menu.

Following are the wealth of programs under submenu program –

## Claws Mail

It is in the internet part of the Application with the help of which you can send or receive messages on your Raspberry Pi computer.

## **Debian Reference**

As we have discussed earlier, the Raspbian version of Linux is the Pi-specific version of the Debian distribution. This icon will guide us how to use Linux on your Raspberry Pi computer.

This is a reference document, which is stored on your SD card and to find this, you need to go through the help section of the Application menu.

To get started with this, first, you need to click the icon and then, click the multi-files link (it is an HTML link) which is at the top of the screen.

## **LibreOffice**

This is the most popular suite of productivity applications. It mainly includes word processing, spreadsheets, and presentations. You can get it from the office section of the Application menu.

## **Mathematica**

Mathematica, under the programming section of the Application menu, is based on the Wolfram programming language. It is used for scientific and technical computing.

## **Minecraft Pi**

We know about the world-building game called Minecraft. Similarly, Minecraft Pi is the Raspberry version of that. You can find it under the Game section of the Application program, and you can also program it by using the Python programming language.

## **Python 2 and Python 3**

Raspberry Pi provides us the Python programming language, which can be found under Programming in the Application menu. We can also use the Thonny IDE (integrated development environment) which provides the Pi users, an alternative way of creating Python programs.

## **Python games**

Raspberry Pi has games such as Reversi, Four in a Row, a sliding puzzle game as well as a snake game. These all are built in Python programming language and can be found in the Game section of the Application menu.



## **Scratch**

Raspberry Pi foundation provides us a simple programming language called Scratch, which is approachable for the peoples of all ages. You can use it to create games and animations. It can also be used to manage electronic projects. You can find it under the Programming section of the Application menu.

## **Sense HAT emulator**

As the name implies, it has some built-in sensors that can be used for creating experiments and other projects. It is an add-on for the Raspberry Pi users, which can be found under the Programming section of the Application menu.

## **Shutdown**

Shutdown, a top-level option in the Application menu, can be used for switching off your Raspberry Pi, before you remove the power. With this, we will also get the options to log out as well as restart your Pi computer.

## **Sonic Pi**

It is another programming language provided by Raspberry Pi foundation which is mainly used for creating music. You can also find it under the Programming section of the Application menu.

## **Terminal**

Terminal is a window that let us issue the instructions from a command line without leaving your PIXEL desktop environment. There are two ways through which you can reach the terminal window. One is to get it in the Accessories part of the Programs menu and other is to use the button on the taskbar.

## **Wolfram**

Wolfram is a programming language provided by Raspberry Pi foundation. It aims to incorporate knowledge, so that the programmers can get results quickly. You can get more information about this at [www.wolfram.com/language](http://www.wolfram.com/language). It is under the Programming section of the Application menu.

## **Running programs**

Even after installing, some of the programs won't appear on the Application menu. You can use the Run option to run those programs.

Follow the below given steps –

**Step 1** – First, we need to open the Application menu. For this, click the icon at top left of the desktop.

**Step 2** – Now, we need to select the Run option from this menu.

**Step 3** – Run option will give you a dialog box. You can type the name of the program, which you want to open, and then press Enter.

## **Close and Rearrange Programs**

The controls for closing and rearranging the programs on the PIXEL desktop environment is like the ones in MS Windows. These controls enable us to close as well as resize (minimize and maximize) the programs.

You can find these controls in the top right. They are explained below –

- ❖ X button – It is used for closing the programs/applications.
- ❖ Maximize button – As the name implies, this button will enlarge a particular application. Once used, the application will fill the screen.
- ❖ Minimize button – As the name implies, this button will reduce a particular application. It will hide the program from view but does not stop it from executing/running. We can return to the program by clicking the name of the program on the taskbar.

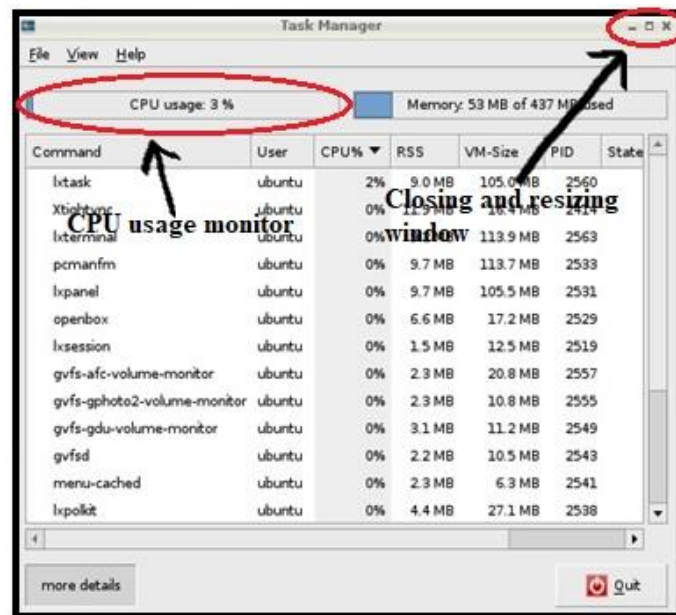
## **Raspberry Pi - PIXEL Desktop Environment**

Let us learn about the PIXEL desktop environment in Raspberry Pi. First, we will understand what a task manager is.

### **Task Manager**

Sometimes, it may happen that your Raspberry Pi computer does not seem to be responding. But, there is nothing to worry about. This happens when the computer might be quite busy.

The diagram below shows the task manager.



In the top right side, you can see the CPU usage monitor, which will tell you how heavily your Pi's processor is being used. Moreover, in the top left side of this taskbar, we have three buttons, which are collectively called as the closing and resizing window.

There are two options to open the task manager, which are as follows –

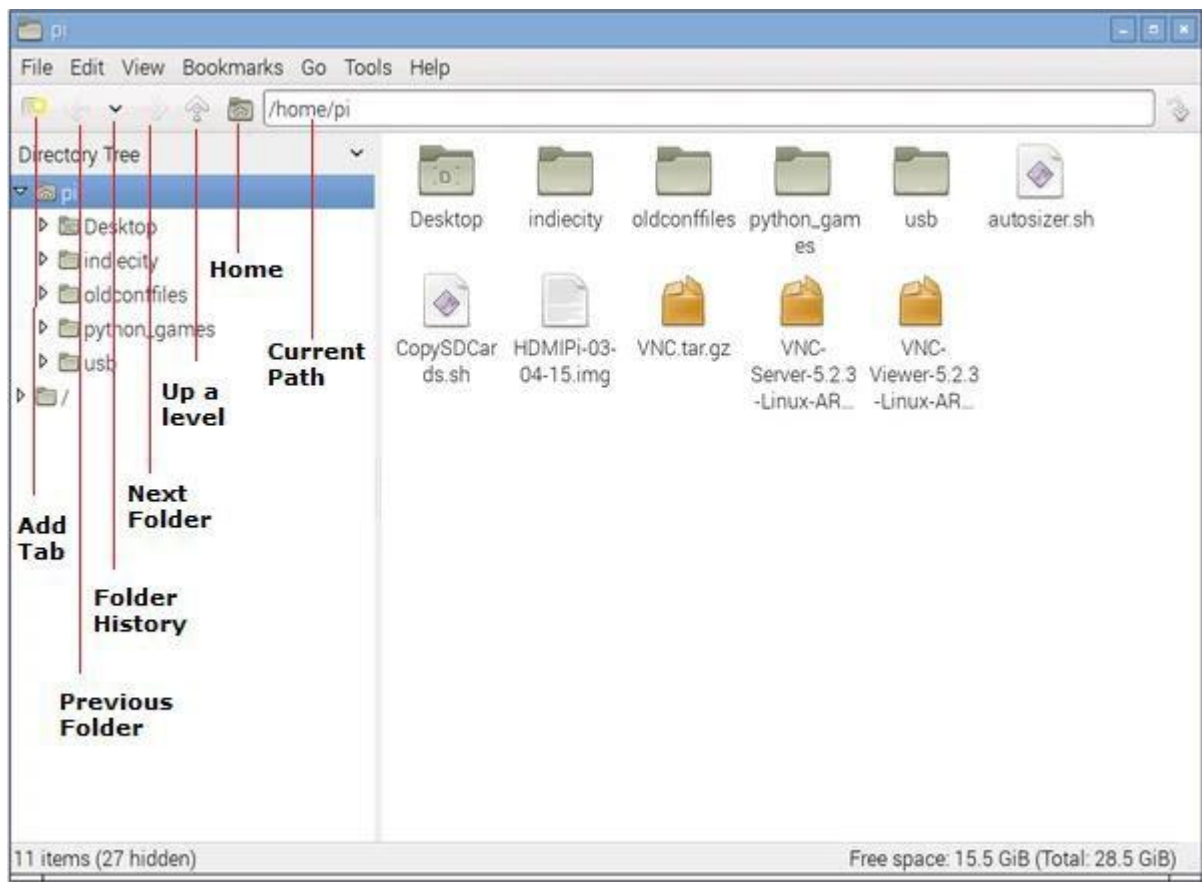
- ❖ Go to the Accessories folder on the Application menu.
- ❖ Use the shortcut key, which is by holding down Ctrl and ALT keys and then pressing the Delete key.

If any of the program is responding and you want to terminate it, just right-click in the task list. The menu will appear, and you can choose the Term from it. This option will give a chance to shut down the program safely.

On the other hand, we can also use Kill, but, this option will terminate the program immediately with loss of data.

## File Manager

It is easy to manage your files in a PIXEL desktop environment rather than using the command line. Refer the screen given below for the file manager –



With the help of file management, we can browse, copy, rename or delete the files on our Raspberry Pi or other connected storage devices.

You have two options to start the file manager, which are as follows –

- ✓ Click the button on the left of the desktop.
- ✓ Go to system tools under the Application menu.

### Navigate File Manager

There is an icon bar, having useful shortcuts, under the file manager's menu bar.

Let us navigate the file manager and understand the icons that come under it.

## **Add Tab**

Suppose, if you want to work in two folders at the same time. For example, copy files from one folder to another. Then, you need to quickly switch between those two folders. Tabs become handy for this purpose.

It enables us to have two different folders open at the same time, so that we can simply click them to switch between them. You can close the tab by clicking the cross (X) icon on the tab.

## **Previous folder**

Previous folder button, as the name implies, takes you back to the last folder, which we have accessed on that tab. It works a bit like a web browser's back button.

## **Next folder**

Next folder button, as the name implies, takes us to a folder, where we have visited after the folder on which we have been working. We will end up where we started, if we will first click the Previous folder button and then the Next folder button.

## **Folder History**

The Folder history button, as the name implies, will open a menu having the folders we have visited.

## **Up a level**

There can be parent and child folders in your Raspberry Pi desktop. For example, the Desktop folder is inside the Pi folder. Hence, the Pi folder will be the parent folder and Desktop folder will be the child folder. The up a level button will take you to the parent folder.

## **Home**

The home button, as the name implies, takes us back to the Pi folder.

## **Path**

Path, as you have seen in the web browser's URL bar, is the text description if it is the location of the folder, we are working with. It also includes the list of the folders, which are above it.

## **Cut, copy, move files and folders**

The file manager in the PIXEL desktop environment makes it easy to move your files and folders from one place to another. It also makes it easy to cut, copy, and paste your files and folders.

You just need to right-click a file or folder of your choice and a menu will appear. This menu has the following options –

- ❖ Renaming the file.
- ❖ Moving the file to the wastebasket.
- ❖ Cut or copy the files.

Now, if you want to cut the file or folder, right-click on that and choose option cut. After that, right-click an empty space, where you want to paste that folder. From the menu that appears, select paste and your file or folder will be pasted at that empty space.

Likewise, if you want to copy the file or folder, you need to choose option copy from the right-click menu and then, paste wherever you want. It will create a duplicate file or folder.

## **Multiple files and folders**

If you want to select more than one file at a time, there are following methods to do it –

### **First method**

You need to hold the CTRL key and then, click on every file you want to select.

### **Second method**

For selecting a group of consecutive icons, you first need to click the first icon, press the SHIFT key, and then click the last icon.

### **Third method**

In this method, you need to click the mouse on the background of the file manager. Now while holding the button, you need to loop all the files which you want to select.

## **Moving the files**

Now for moving these files, you have the following methods –

- ❖ Once you have selected the files, you can drag these files into a different folder.
- ❖ Otherwise, you can choose the option of cut or copy the whole group of files by right-clicking one of the selected files.

### **Keyboard Shortcuts**

Like MS windows, you can also use some shortcuts in PIXEL as follows –

- Ctrl+A – To select all the files and folders.
- Ctrl+C – To copy the files and folders.
- Ctrl+V – To paste the files and folders.
- Ctrl+X – To cut the files and folders.

### **Organize files in folders**

To easily manage your files, you can organize them in a folder. It is easy to create a new folder.

Follow the below given steps –

- ❖ First, select and go to the location, typically your pi folder, where you want to create a new folder.
- ❖ Now, right-click a blank space in the File Manager and click on the option Create New from menu.
- ❖ Now another menu will appear, and you need to click Folder from that menu. You will now be prompted to enter a name. Enter the name you want to give to the new folder and click OK to confirm.

Other method to create a New Folder is to click the File menu at the top left of the File Manager and find Create New. With these methods, you can also create empty files.

### **Delete files and folders**

If you want to delete a single file or folder, you can right-click that in the File Manager. For the menu, you need to choose the option Move to Wastebasket.

On the other hand, if you want to delete more than one file or folder, you can select all of them as we did before, and choose the option Move to Wastebasket right-click menu.

You can also use the keyboard Delete button to send the selected files to the wastebasket.

## **Sorting the files**

You can sort your files in Raspberry Pi by name, size, file type, modification time, etc. For this, you again need to right-click the empty space in the right pane of the file manager. A menu will appear and you need to select the option to change how the files are sorted.

You can also change how your files are displayed in the file manager. For this, you need to use the View menu on the Menu bar at the top of the File Manager.

The View menu will give us the following four ways to display our files and folders –

### **Icon view**

It is the default option used by the File manager. It strikes a good balance between the size of each icon and number of files, we can see at one time.

### **Thumbnail view**

Another view option is thumbnail view, which is mostly used in a folder of images. It enlarges the preview.

### **Compact view**

As the name implies, the compact view lists the files and folders in columns and this is done with a small icon and filename. It helps us to view as many files as possible at a time.

### **Detailed view**

As the name implies, this view reveals detailed information like short description, size, last modification date, etc., about the file.

Now, let us continue learning about the other important aspects with regards to the PIXEL desktop environment in Raspberry Pi.

## **Browsing the Web**

Raspberry Pi gives us the choice of four browsers for browsing the web, namely, Chromium, Dillo, Netsurf and Epiphany. You can type the name of the browsers into the run option on the Application menu and it will appear.



## **Chromium Browser**

It is the recommended browser. Apart from the Run option, you can also access it by clicking the Web browser button (the Globe icon) in the top left of the screen.

The layout of Chromium browser is quite similar to other browsers. It has a thin toolbar and provides a maximum screen to the page which you are accessing. It provides the user with the facility for ad-blocker to strip advertising. You can also change the settings for the same.

## **Dillo Browser**

This browser is fast. Hence, it is a good choice for those users who have a slow internet connection and have problems related to accessing mainly the text information. It does not support Javascript and cannot handle the sophisticated layout instruction.

This is the reason that the web pages look different than the ones intended on it. This browser provides the users with an option to switch off the images from the Tool menu to speed up the downloads of complex pages.

## **Netsurf**

This web browser is capable of handling more sophisticated layouts than Dillo web browser. But like Dillo, netsurf also lacks support for Javascript. Hence, the websites (including Facebook) that require Javascript won't work on Netsurf.

## **Epiphany**

It supports Javascript and was the recommended browser before Chromium. Epiphany browser is optimized for the Raspberry Pi but, might be noticeably slower than what we are used to.

## **Claws Mail**

Raspberry Pi provides us an open-source email program called Claws Mail. It is preinstalled and you can find it in the internet category of the Application menu.

Following are some prerequisites, if you want to use email on Raspberry Pi –

- ❖ For sending emails, you need to know the details of the server. You can find this information on the website of your email provider.

- ❖ Your email user ID and password. This should be the same as you use when logging on with webmail.

## **Sending and Receiving Emails**

Follow the below given steps to use Claws mail for sending and receiving emails –

- ❖ First, you need to add an account from the configuration wizard of Claws mail. Apart from adding a new account, you can also edit the account settings, delete an account by using the configuration menu.
- ❖ Once you are done with configuration, go to top left and click the Get mail button. It will show your mail folder in the left and messages on the right at the top.
- ❖ To read the messages, you can use two methods. One is to use the message preview pane at the bottom right and other is to double-click on a message to open in its own window.
- ❖ For composing a new message, replying, and forwarding a message, there is a menu bar across the top of Claws mail.

## **Image Viewer**

If you want to look at your Digital images and work with them in Raspberry Pi, PIXEL provides us with the Image Viewer. You can find it in among the accessories on the Application Menu.

### **Toolbar buttons**

You will see a toolbar underneath the picture opened in the Image Viewer.

Following are the buttons on that toolbar –

#### **Previous**

As the name implies, with this button, you will go to the previous photo in that folder. Any unsaved changes would be lost.

#### **Next**

As the name implies, with this button, you will go to the next photo in that folder. Any unsaved changes would be lost.

### **Start Slide show**

This button will begin a slide show of all the photos in that folder. Predefined interval between two photos is 5 seconds. Image Viewer gives us an option to change it in preferences. Keyboard shortcut for starting slide show is key W.

### **Zoom Out**

This button will reduce the magnification of the image. Keyboard shortcut for Zoom Out is the Minus (-) key.

### **Zoom In**

This button will increase the magnification of the image. Keyboard shortcut for Zoom In is the Plus sign (+) key.

### **Fit Image to Window**

It will shrink a large image to make it fit in the Image Viewer window. Its keyboard shortcut is key F.

### **Go to Original Size**

This button will reset all the zooming by showing an image at its original size. Its keyboard shortcut is key G.

### **Full Screen**

As the name implies, this button will expand an image to fill the monitor. With the use of this button, you will lose the Image Viewer controls.

### **Rotate Left**

It will rotate the image 90 degrees counterclockwise. Keyboard shortcut for Rotate Left is key L.

### **Rotate Right**

It will rotate the image 90 degrees clockwise. Keyboard shortcut for Rotate Right is key R.

**Flip Horizontally**

This button will mirror the image horizontally. Keyboard shortcut for Flip Horizontally is key H.

**Flip Vertically**

This button will mirror the image vertically i.e., turns an image upside down. Keyboard shortcut for Flip Vertically is key V.

**Open File**

It will open a new image file. You can also open an image from a folder in File Manager by using the Drag-and-Drop option on Image Viewer.

**Save File**

It will save an image with changes which you have done. It will replace the original file. The keyboard shortcut key is S.

**Save File as**

It will save an image, with the changes which you have done, with a new filename. It will not replace the original file.

**Delete**

It will delete an image from the storage device. If you use this button, an image will be deleted permanently and won't be recovered.

**Preferences**

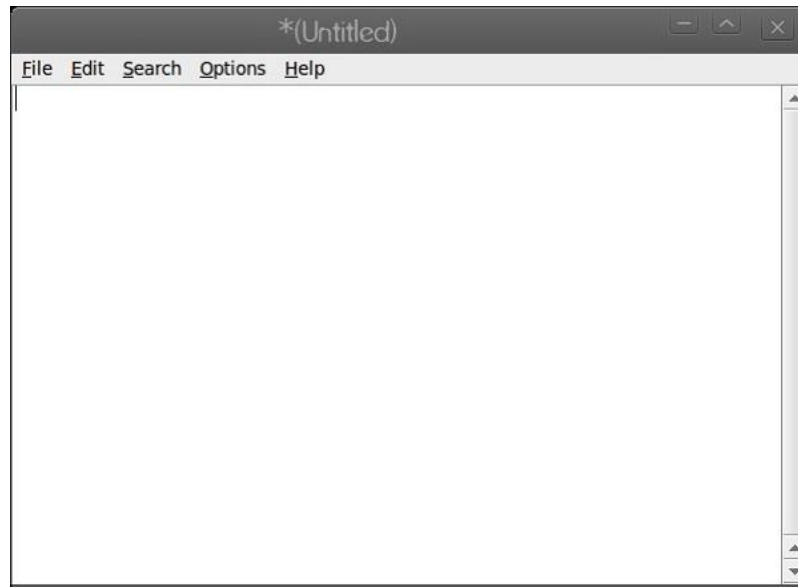
This button holds the settings, which you can change for the Image Viewer. It permits you to customize the settings as per your needs.

**Exit Image Viewer**

As the name implies, it will close the Image Viewer application. We can also use the close button (X) which is in the top right.

## Text Editor

PIXEL has a simple text editor called Leafpad. You can find it by clicking the Text Editor in the Accessories part of the Application Menu. Leafpad text editor is good for writing and word processing but not ideal for creating the print-ready documents.



## Leafpad

The menu on Leafpad consists of the following buttons –

### File Menu

You can use this menu to start the new documents as well as to open, save, and print the files. It has a quit option, which we can use to close the text editor.

### Edit Menu

The Edit menu gives you the tools to do the following tasks –

- ❖ Undoing your work.
- ❖ Redoing your work.
- ❖ Cutting the work.
- ❖ Copying your work.
- ❖ Pasting the work.
- ❖ Deleting the work.
- ❖ Selecting all the text.

It uses the same shortcuts as MS Windows, as mentioned below –

- ❖ Ctrl+C – To copy the work.
- ❖ Ctrl+V – To paste the work.
- ❖ Ctrl+X – To cut the work.
- ❖ Ctrl+A – To select all the text.

### **Search Menu**

This menu gives us the following options –

- ❖ To find a particular word or phrase.
- ❖ Jump to a particular line in a document.
- ❖ Replace a chosen word or phrase with an alternative.

### **Option Menu**

This menu gives us the following options –

- ❖ To change the font.
- ❖ Switch on word wrap.
- ❖ Switch on line numbers.

### **Customize your Desktop**

You can change the look and feel of your desktop and make it easier to use by doing some changes. The options to customize your desktop are under the Preferences section of the Application menu.

With these options we can do the following –

- ❖ Change the picture used as a backdrop i.e., wallpaper.
- ❖ Change desktop color, if not using wallpaper.
- ❖ Change the color of icon descriptions i.e., the text color.

### **Install New Applications**

Although, we can use the command line to discover and install new software, but, there is also a friendly menu in the PIXEL desktop environment. For that menu, we need to go to the Preferences option and click on ADD/REMOVE software.

This menu has the following parts that helps us to find and install new applications –

## **Search box**

The search box is at the top left. Here, you can enter the name of a program you are looking for and it will show you the options.

## **Main pane**

It shows us the packages. The already installed packages will be checked and would be in bold. Tick the box titled decide it, if you want to install that package.

After choosing your software, you need to click the Ok button to install and remove the applications. It will prompt for entering the password.

## **Back up the data**

To back up your data, you can use File manager to copy them to a USB key or MicroSD card. Raspberry Pi provides us an application called the SD Card Copier application for copying data.

You can also use the shell commands.

### *Raspberry Pi - Linux Shell*

The Shell, called Bash in Raspberry Pi, is the text-based way of issuing instructions to your Pi board. In this chapter, let us learn about the Linux shell in Raspberry Pi. First, we will understand how to open a shell window.

## **Open Shell Window**

You can open a shell window by using one of the two following ways –

- There is a Terminal icon, having a >\_ prompt, on the top of the screen. Click on it and you will get a shell window.
- Another way is to use the Accessories section of the Application menu. You can find the Terminal there.

Both of the above approaches will open a shell window on the desktop.

## Understanding the Prompt

The prompt looks like as follows –

```
pi@raspberrypi ~ $
```

It contains lots of information. Let us see the various bits –

**pi**

It represents the name of that user who logged in.

**raspberrypi**

It represents the hostname of the machine i.e.; the name other computers use to identify when connecting to it.

The tilde symbol (~)

The tilde symbol tells the user which directory they are looking at. The presence of this horizontal wiggly line is known as home directory and the presence of this symbol shows that we are working in that directory.

The dollar sign (\$)

It represents the presence of the ordinary user and not all-powerful superuser. A # symbol means a superuser.

## List Files and Directories

When you start the shell window, you start in your home directory.

To see the folders and files in your home directory, you need to issue a command which is as follows,

```
pi@raspberrypi ~ $ ls
```

## Output

The output is as follows –

```
Desktop Downloads Pictures python_games Videos
Documents Music Public Templates
```

You can see the files and folders after issuing the ls command.



As we know that Linux is case sensitive hence the commands LS, Ls, ls and lS are all different.

### **Change the directory**

You can see the above output, it's all blue which means these are all directories. We can go to these directories and check which files they contain. The command to change the directory is cd. You need to use the cd command along with the name of the directory which you want to see.

The example for changing the directory in Raspberry Pi is given below –

```
pi@raspberrypi ~ $ cd Pictures
```

### **Find information about files**

The command to find the information about a particular file is file. You need to put the name of the file after the command to check the information about that file.

Check the below example for finding information about the files in Raspberry Pi–

```
pi@raspberrypi ~ /Pictures $ file leekha.png aarav.png
leekha.png: PNG image data, 50 x 85, 8-bit/color RGBA, noninterlaced
aarav.png: PNG image data, 100 x 150, 8-bit/color RGBA, noninterlaced
```

We can also use the file command on directories. It will give us some information about the directories as well –

```
pi@raspberrypi ~ $ file Pictures Desktop
Pictures: directory
Desktop: directory
Parent Directory
```

Earlier, we have used the cd command to change into a directory that is inside the current working directory. But sometimes, we need to go to the parent directory i.e. into the directory which is above the current working directory.

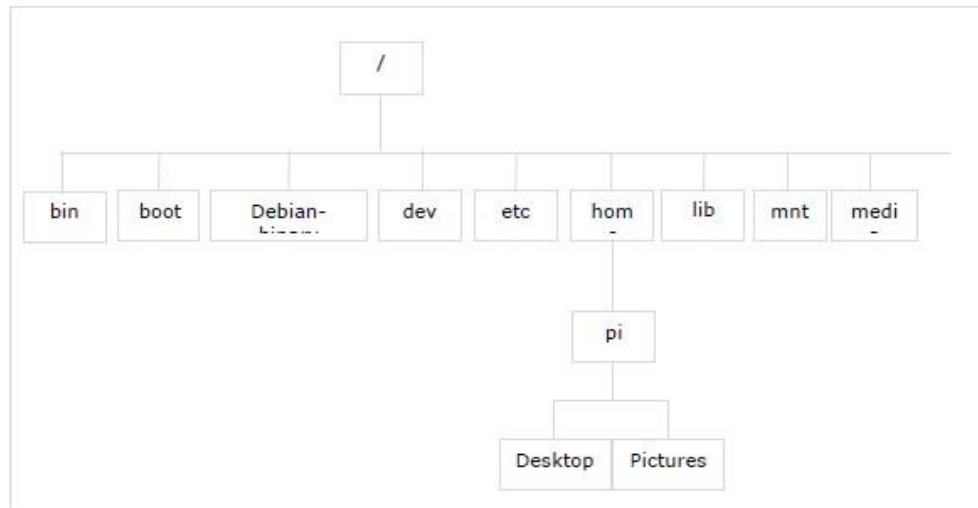
The command for this is cd..(cd with two dots), as given below –

```
pi@raspberrypi ~ /Pictures $ cd..
pi@raspberrypi ~ $
```

The tilde symbol represents your home directory.

## Directory Tree

The following diagram shows the part of the directory tree on your Raspberry Pi computer –



The directories and their uses are as follows –

### **bin**

Bin, short for binaries, contains some small programs that behave like commands in the shell. For example, ls and mkdir.

### **boot**

This directory contains the heart of the OS i.e. the Linux kernel. It also contains the configuration files containing the technical settings for Raspberry Pi computer.

### **dev**

This directory contains a list of devices. For example, the devices like disks and network connections.

### **etc**

This directory is used for various configuration files. These configuration files apply to all the users on the computer.

## **home**

This is the directory where a user can store or write files by default.

## **lib**

The directory contains various libraries that are used by different OS programs.

## **lost+found**

This directory is used, if the file system gets corrupted and recovers partially.

## **media**

You connect a removable storage device such as a USB key and it is automatically recognized. All the details will be stored in the media directory.

## **mnt**

mnt stands for mount and will store all the details of the removable storage devices that we mount ourselves.

## **root**

It is reserved for the use of the root users and we don't have the permission to change into this directory as an ordinary user.

## **Relative and Absolute Paths**

The shell enables the Raspberry Pi users to go straight to the location by specifying a path.

*We have the following two types of paths –*

### **Relative path**

It is a bit like giving the directions to the directory from where the user is now.

### **Absolute path**

On the other hand, an absolute path is like a street address. This path is exactly the same wherever the user is. These paths are measured from the root. Hence, they start with a slash (/).

For example, we know the absolute path to the pi directory is /home/pi.

Now, go straight forward to this directory by using the following command –

```
cd /home/pi
```

If you want to go to the root, you can use the following command –

```
cd /
```

### Advanced Listing Commands

We can use the listing command (ls) to look inside any directory outside the current working directory as follows –

```
pi@raspberrypi ~ $ ls /boot
```

There are several advanced options, which we can use with the ls command.

These options are given in the following table –

| Option | Description                                                                                                                                                                                      |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -l     | This option is l not I and it outputs the results in a single column instead of a row.                                                                                                           |
| -a     | The ls command with this option will display all the files. All the files will also include hidden files.                                                                                        |
| -F     | This option will add a symbol besides a filename. It will do this to indicate the file type. If you use this option, you will notice a / after directories names and a * after executable files. |
| -h     | This option is short for human-readable. It expresses file sizes by using kilobytes, megabytes, and gigabytes.                                                                                   |
| -l     | This option will display the result in the long format. It shows the information about the permissions of files, their last modification date, their size.                                       |
| -m     | This option will list the result as a list separated by commas.                                                                                                                                  |

|    |                                                                                                                                                                              |
|----|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -R | This option is the recursive option. It will also list files and directories in the current working directories, open the subdirectories(if any) and list their results too. |
| -r | It is the reverse option and will display the result in reverse order.                                                                                                       |
| -S | This option will sort the result by their size.                                                                                                                              |
| -t | This option will sort the result as per the date and time they were last modified.                                                                                           |
| -X | This option will sort your result as per the file extension.                                                                                                                 |

Furthermore, we will learn about the other important aspects related to Linux Shell in Raspberry Pi.

### **Long Listing Format**

Long format is one of the most useful formats of the ls command, because it provides us the additional information on the file.

You can use the ls command with the long listing option as follows –

```
pi@raspberrypi ~ $ ls -l
total 65
-rw-r--r-- 1 pi pi 256 Feb 18 22:45 Leekha.txt
drwxr-xr-x 2 pi pi 4096 Jan 25 17:45 Desktop
drwxr-xr-x 5 pi pi 4096 Jan 25 17:50 Documents
drwxr-xr-x 2 pi pi 4096 Jan 25 17:52 Downloads
drwxr-xr-x 2 pi pi 4096 Jan 25 17:53 Music
drwxr-xr-x 2 pi pi 4096 Jan 25 17:45 Pictures
drwxr-xr-x 2 pi pi 4096 Jan 25 17:45 Public
drwxr-xr-x 2 pi pi 4096 Jan 25 17:54 Templates
drwxr-xr-x 2 pi pi 4096 Jan 25 17:54 Videos
```

From the above output, it is very easy to understand that each line relates to one file or directory having its name on the right and the date and time, when it was last modified next to that.

The number 256, 4096 represents the size of the file. You can see some files and directories are having the same size.

The remaining part of this output shows the permissions i.e. who is allowed to use the file and what the user is allowed to do with that file or directory.

## **Permissions**

The permissions on a file are divided in the following three categories –

### **Owner**

It is the person who created the file. This permission consists of the things the file owner can do.

### **Group**

These are the people who belong to a group that has the permission to use the file. This permission consists of the things which the group owners can do.

### **World**

These are known as the world permissions i.e. the things that everyone can do with that file or directory.

In Raspberry Pi, we have two main types of files. One is regular files which have a hyphen (-) and others are directories having a d.

## **Types of Permissions**

Now let us understand the different types of permissions the owner, group and world have respectively –

- ✓ Read permission – This permission gives the user ability to open and look at the contents of a file or to list a directory.

- ✓ Write permission – This permission gives the user the ability to change the content of a file. It allows the user to create or delete the files in a directory.
- ✓ Execute permission – This permission gives the user an ability to treat a file as a program and run it. It also gives permission to enter a directory using the cd command.

## **Less Command**

The ls command deluges with the information that you cannot even notice sometimes, because it flies past our eyes faster than we understand or see it. To avoid this or solve this problem, we can use a command called less.

This command will take our listing and enable us to page through it and that is one screen at a time. To use this command, we need to use a | (pipe character) after the listing (ls) command.

The example of less command in Raspberry Pi is given below –

```
ls -RFX | less
```

The less command can also be used to view the content of a text file.

For this, we need to provide the filename as an argument, as given below –

```
less /boot/config.txt
```

## **Speed up the use of Shell**

Here we will be learning few tricks to speed up the use of the shell –

- If you want to retype a command, then you can save retyping it because, shell keeps the record of history i.e. the commands you entered previously.
- In case if you want to reuse your last command, you just need to use two exclamation marks and press enter.
- You can also bring back the previous commands in order by tapping the up arrow.
- Similarly, you can also move through your history of commands in another direction by tapping the down arrow.
- The shell also guesses what the user wants to type and it also automatically completes it for us.

## Create File with Redirection

Redirecting files means, you can send the results from a command to a file instead of sending the results to the screen. For this, we need to use a > (greater-than) sign along with the file name, which we would like to send the output to.

The example of creating file by using redirection in Raspberry Pi is given below –

```
ls > ~/gaurav.txt
```

There are other commands too and with their help, we can display the content online. These commands are explained below –

echo command

The echo command, as the name implies, will display on screen whatever we write after it. The best use of this command is to solve mathematical problems. You need to put the expression between two pairs of brackets and put a dollar sign in front.

The example of echo command is given below –

```
echo $((5*5))
```

### date command

The date command, as the name implies, will display on screen the current date and time.

### cal command

The cal command (cal stands for calculator) will display the current month's calendar with today highlighted. With the help of option -y, you can see the whole year calendar.

## Create and Remove Directories

Here, we will understand how to create and remove directories in Raspberry Pi. Let us begin by learning about creating directories.

### Create Directories

The command to create a directory under your home directory is mkdir.

In the below example, we will be creating a directory named AI\_Python –

```
mkdir AI_Python
```

You can also use one command to create several directories as follows –



```
pi@raspberrypi ~ $ mkdir AI_Python Machine_Learning Tutorialspoint
```

```
pi@raspberrypi ~ $ ls
```

Downloads AI\_Python Machine\_Learning Tutorialspoint Desktop Pictures Documents Public

### **Remove directories**

If you want to remove an empty directory, you can use the command `rmdir` as follows –

```
pi@raspberrypi ~ $ rmdir AI_Python
```

On the other hand, if you want to remove non-empty directories, you need to use the command `rm -R` as follows –

```
pi@raspberrypi ~ $ rm -R Machine_Learning
```

### **Delete Files**

We can use the `rm` command to delete a file.

The syntax for deleting a file would be as follows –

```
rm options filename
```

In an example given below, we will be deleting a text file named `leekha.txt` –

```
pi@raspberrypi ~ $ rm leekha.txt
```

Like `mkdir`, the `rm` command will not tell us what it is doing.

To know its function, we need to use the verbose(`-v`) option as follows –

```
pi@raspberrypi ~ $ rm -v leekha.txt
```

```
removed 'leekha.txt'
```

We can also delete more than one file at a time as follows –

```
pi@raspberrypi ~ $ rm -v leekha.txt gaurav.txt aarav.txt
```

```
removed 'leekha.txt'
```

```
removed 'gaurav.txt'
```

```
removed 'aarav.txt'
```

### **Raspberry Pi Wildcards**

A directory contains a lot of files with the similar filenames and if you want to delete a group of such files, you don't need to repeat the command by typing out each filename. In shell, wildcards will do this job for us.

Following table provides us a quick reference to the wildcards, which we can use in Raspberry Pi –

| Wildcard | Meaning                                                                         | Example       | Description                                                                                                        |
|----------|---------------------------------------------------------------------------------|---------------|--------------------------------------------------------------------------------------------------------------------|
| ?        | It means any single character.                                                  | pic?.jpg      | The example means that the files start with a pic and have exactly one character after it before extension starts. |
| *        | It means any number of characters.                                              | *pic*         | The example means that any files that have the word pic in their filename.                                         |
| [...]    | This wildcard will match any one of the characters in brackets.                 | [gla]*        | The example means that all the files that start with the letter g, l or a.                                         |
| [^...]   | This wildcard will match any single character that is not between the brackets. | [^gla]*       | The example means that any files that do not start with the letter g, l or a.                                      |
| [a-z]    | This wildcard will match any single character in the range specified.           | [x-z]*.png    | The example means that any files that start with a letter x, y or z and end with the .png extension.               |
| [0-9]    | This wildcard will match any single character in the range specified.           | Pic[1-5]*.png | The example means that it will match pic1.png, pic2.png, pic3.png, pic4.png, and pic5.png.                         |

The below given example will remove all the files starting with letters lee,

```
rm -vi lee*
```

## **Copy Files**

Copying files is one of the fundamentals things we would like to do.

The command for this is `cp`, which can be used as follows –

```
cp [options] copy_from copy_to
```

Here, we need to replace `copy_from` with the file you want to copy and `copy_to` for where you want to copy it.

### ***Example***

Let us see an example of using the command to copy the respective file.

Suppose, if you want to copy the file `leekha.txt` from the `/desktop` directory to the home directory, you can use the `cp` command as follows –

```
cp /Desktop/leekha.txt ~
```

We can also specify a path to an existing folder to send the file to as follows –

```
cp /Desktop/leekha.txt ~/doc/
```

## **Move Files**

Rather than making a copy of the file, if you want to move it from one place to another then, you can use the `mv` command as follows –

```
mv ~/Desktop/leekha.txt ~/Documents
```

The above command will move the file named `leekha.txt` from `Desktop` directory to the `Documents` directory. Both of these directories are in the home directory.

## **Reboot Raspberry Pi**

With the help of following command, we can reboot our Raspberry Pi without disconnecting and reconnecting the power –

```
sudo reboot
```

## **Shutdown Raspberry Pi**

With the help of following command, we can safely turn off our Raspberry Pi –

```
sudo halt
```

## 8.9. Raspberry Pi - Managing Software:

We have discussed the simple menu based ADD/REMOVE software tab under Preferences for installing software. It is one of the easiest ways to manage and install software on your Raspberry Pi.

But here, we will be discussing how we can use the command line to install software in Raspberry Pi.

To install software, we require the authority of the root user or superuser but, sometimes that leaves our Raspberry Pi computer files vulnerable, including to any malicious software that might get in.

We can use sudo instead of a root account. Putting sudo before a command will indicate that one wants to carry it with the authority of the root user.

### Update Cache Memory

If you want to install software on your Raspberry Pi, you first need to update the cache memory. It is the list of the packages, which the package manager knows about.

Use the following command to update the cache memory –

```
sudo apt-get update
```

### Find the Software

To find the package name or software, we need to use the package manager cache. In Linux terminology it is the apt, cache.

It contains an index of all the packages available for install. It collects information of the software package and, also used to search for the available packages which are ready for installation on Raspberry Pi.

With the help of following command, we can search the required software –

```
sudo apt-cache search pkgname
```

Suppose if you want to search the games packages, you can use the command as follows

–

```
sudo apt-cache search game | less
```

The list might be long. Hence, we have used less.

And suppose, if you want to find the package name for a particular game say chess, you can give the title in the command as follows –

```
sudo apt-cache search chess
```

This command will search for all the packages with name chess.

## **Install Software**

Once you finish the searching, you can now install the software. For searching, you used apt-cache. However, for installing, you need to use apt-get command.

The command will download the specific package from the internet and install it. It will also install other dependencies.

For example, if we want to install the chess game say 3dchess, then the command will be as follows –

```
sudo apt-get install 3dchess
```

## **Run the Software**

Following are the two ways to run a particular program in Raspberry Pi –

### **From command line**

You can run some programs directly from the command line. You need to type the name of the program as follows –

```
3dchess
```

It will directly run the program.

### **From Application menu**

Another way is to use the Application menu. After installing, you can find the application in the Application menu.

In Raspberry Pi, most of the end-user applications require the X-server. It means they need the Desktop environment to run them.

## **Upgrade Software**

You can use the package manager to keep your software.

Following is the command with the help of which we can update all our software –

```
sudo apt-get upgrade
```

On the other hand, if you want to update just one application, you can do it by issuing its install command again.

For example, we have installed the chess game above, now enter the same again –

```
sudo apt-get install 3dchess
```

The above command will prompt the apt to check for any updates of the package and install them. If it finds no updates, it will tell us that we are running the latest version of the software.

## **Remove Software**

You can also use the package manager to remove software from your Raspberry Pi computer.

You can use the following command to remove the software –

```
sudo apt-get remove 3dchess
```

The above command will remove the 3dchess package, but it will leave some traces of the application. These traces might include user files and any files containing settings.

You can also completely remove the application by using the following command –

```
sudo apt-get purge 3dchess
```

## **What software is installed?**

You can use the following command to find out what software is installed on your Raspberry Pi computer –

```
dpkg --get-selections
```

With the help of following command, you can search for a specific package –

```
dpkg --get-architecture
```

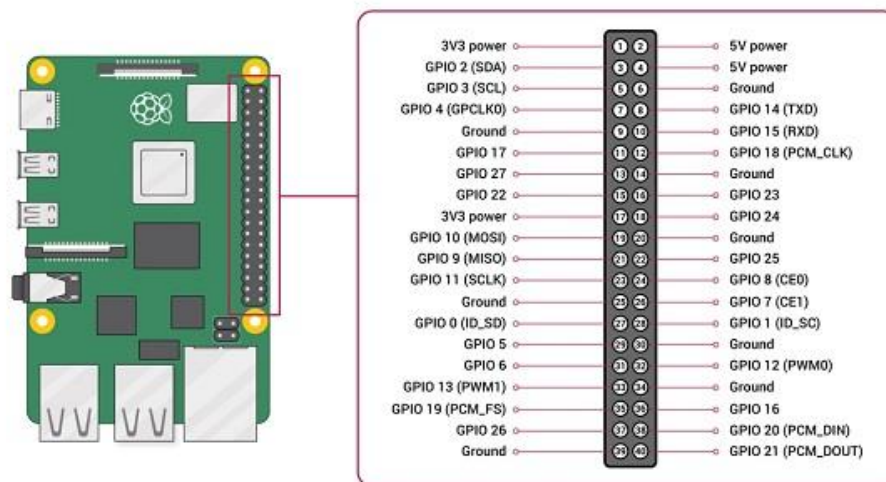
## 8.10. Raspberry Pi - GPIO Connector:

Here, we will learn about the GPIO (general-purpose input output) connector in Raspberry Pi.

### GPIO Pinout

One of the powerful features of the Raspberry Pi is the row of GPIO (general-purpose input output) pins and the GPIO Pinout is an interactive reference to these GPIO pins.

Following diagram shows a 40-pin GPIO header, which is found on all the current Raspberry Pi boards –



The source of the diagram is [www.raspberrypi.org](http://www.raspberrypi.org)

### Voltages

From the above diagram, we can see that there are two 5V pins and two 3V3 pins on the board. It also has several ground pins (0V). All these pins are unconfigurable.

### Outputs

A GPIO pin can be designated as an output pin. The pin set as output pin can be set to 3V3(high) or 0V(low).

### Inputs

A GPIO pin can be designated as an input pin. The pin set as input pin can be read as 3V3(high) or 0V(low). You can use internal pull-up or pull-down resistors.

You can see in the above diagram, GPIO2 and GPIO3 pins have fixed pull-up resistors but for the other pins, you can configure it in software.

### **Alternative Functions**

GPIO pins can be used with a variety of alternative functions. Among them, some are available on all pins and others on specific pins.

### **PWM: Pulse-width modulation**

Software PWM are available on all the pins whereas Hardware PWM are available on GPIO12, GPIO13, GPIO18, and GPIO19.

### **SPI: Serial Peripheral Interface**

The SPI are available on the following –

SPI0: MOSI (GPIO10); MISO (GPIO9); SCLK (GPIO11); CE0 (GPIO8), CE1 (GPIO7)

SPI1: MOSI (GPIO20); MISO (GPIO19); SCLK (GPIO21); CE0 (GPIO18); CE1 (GPIO17); CE2 (GPIO16)

### **I2C: Inter-integrated Circuit**

The I2C are available on the following –

Data: (GPIO2); Clock (GPIO3)

EEPROM Data: (GPIO0); EEPROM Clock (GPIO1)

### **Serial**

The serial function is available at the following –

TX(GPIO14)

RX(GPIO15)

### **Connect GPIO to Raspberry Pi**

Following are some simple rules to reduce the risk of damaging your Raspberry Pi board, while using the GPIO connector –

- ❖ Do not try to put more than 3.3V on any GPIO pin.
- ❖ Do not try to draw more than 3mA per output. Although, you can draw more but to increase the life of your Pi Board, you should restrict upto 3mA.



- ❖ You should not poke the GPIO connector with a screwdriver when the Raspberry Pi board is powered up.
- ❖ 5V power is enough for your Raspberry Pi. Don't try to provide more power than that.
- ❖ You should not try to draw more than a total of 50mA from the 3.3V supply pins.

## **Output of GPIO pins**

To set up the output of GPIO pins and to read the input values by using Python, you need to install RPi.GPIO python library.

### **Install RPi.GPIO python library**

To install RPi.GPIO python library, type the following commands on terminal window of your Raspberry Pi –

```
sudo apt-get install python-dev
sudo apt-get install python-rpi.gpio
```

Almost all the latest versions of distributions have RPi.GPIO already installed. In that case, the above commands will update it to the latest version.

## **I2C Device**

Let us check how we can make I2C work with Raspberry Pi.

### ***Case 1: Using Adafruit Occidentalis 0.2 or later***

In case, if you are using Adafruit Occidentalis, you don't need to do anything. Because, this distribution is preconfigured with I2C support.

### ***Case 2: Using Raspbian***

In case if you are using Raspbian, you need to do the following configuration changes –

First, edit the file /etc/modules by using the following command –

```
sudo nano /etc/modules
```

Now, we need to add the following lines to the end of this file –

```
i2c-bcm2708
i2c-dev
```

Next, we need to edit the file named `/etc/modprobe.d/raspi-blacklist.conf` and comment out the following line by adding a `#` –

```
blacklist i2c-bcm2708
#blacklist i2c-bcm2708
```

Once done, install the Python I2C library by using the following command –

```
sudo apt-get install python-smbus
```

Now, reboot your Raspberry Pi and it will be ready for I2C.

### **Find I2C Address**

There is an I2C device attached to Raspberry Pi computer and you want to know its address.

For this, we need to install `i2c-tools` as follows –

```
sudo apt-get install i2c-tools
```

Once done, attach your I2C device to your Raspberry Pi board and run the following command –

```
sudo i2cdetect -y 1
```

Here, we need to take care about the following two things –

- First, if you have newer distributions then, it is quite possible that it has `i2c-tools` already installed.
- Second, if you have an older version 1 board in use, change 1 to 0 in the above code line.

### **Serial Peripheral Interface (SPI)**

Let us check how we can use SPI (serial peripheral interface) bus with Raspberry Pi.

#### ***Case 1: Using Adafruit Occidentalis 0.2 or later***

In case if you are using Adafruit Occidentalis, you don't need to do anything because this distribution is preconfigured with SPI support.

#### ***Case 2: Using Raspbian***

In case if you are using Raspbian, you need to do the following configuration changes –

First, edit the file /etc/modules by using the following command –

```
sudo nano /etc/modules
```

Now, we need to add the following lines to the end of this file –

```
spidev
```

Next, we need to edit the file named /etc/modprobe.d/raspi-blacklist.conf and comment out the following line by adding a # –

```
blacklist spi-bcm2708
#blacklist spi-bcm2708
```

Once done, install the Python library to carry out communication from a Python program by using the following command –

```
cd ~
sudo apt-get install python-dev
git clone git://github.com/doceme/py-spidev
cd py-spidev/
sudo python setup.py install
```

Now, reboot your Raspberry Pi and it will be ready for SPI.

## **Serial Port**

Suppose if you want to use the serial port i.e. Rx and Tx pins on your Raspberry Pi board but, it is used by Linux OS as a console connection. To disable this, we need to comment out a line in a file named /etc/inittab.

Open this file by using the following code line –

```
sudo nano /etc/inittab
```

Now, find the following line by scrolling down at the end of this file –

```
T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Now, we need to use a # to comment out this line –

```
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Now, save this file and reboot your Raspberry Pi.

## **Access serial port from Python**

We can also use the serial port i.e. Rx and Tx pins on Raspberry Pi board by using Python. For this, we need to install PySerial library as follows –

```
sudo apt-get install python-serial
```

It is recommended to disable (as we did above) the Raspberry Pi's serial console before using PySerial.

## **Test the serial port**

Once you start using the serial ports, you may want to send and receive the serial commands from a Terminal session. For this, we need to install Minicom as follows –

```
sudo apt-get install minicom
```

It is recommended to disable (as we did above) the Raspberry Pi's serial console before using Minicom.

As now Minicom is installed, we can start a serial communication session with a serial device connected to the RXD and TXD pins of the GPIO connector by using the following command –

```
minicom -b 9600 -o -D /dev/ttyAMA0
```

Here, in the above command, after -b is the baud rate and after -D is the serial port. We should use the same baud rate as set on the service we are communicating with.

## **8.11. Raspberry Pi - Add-on Boards:**

There are ready-made boards with all sorts of components built on. Many companies have produced such boards. Such boards come with sample code to show us how to use them.

### **Types of ready-made boards**

We have two types of ready-made boards, Raspberry Pi, which are as follows –

- ❖ The boards which are designed for making it easy to get access to the GPIO pins.
- ❖ The boards with components that have already soldered up.

### **Styles of Ready-made Boards**

The ready-made boards come in following three styles –

## **Separate boards**

These kinds of ready-made boards connect to the GPIO pins via a ribbon cable or your own wires.

## **Shield or plates**

These kinds of ready-made boards plug into all the GPIO pins and cover most of the area of the Raspberry Pi board.

## **HATs (Hardware Attached on Top)**

These kinds of ready-made boards are like the shields or plates but contain an additional identification. Sometimes, it may contain a software so that the Raspberry Pi can read what they are on start-up and install some software and prepare GPIO pins automatically.

## **Various Boards**

Since the introduction of the Raspberry Pi computer, new boards are constantly being developed and produced.

Some of the boards are as follows –

### **The Sense HAT**

The Sense HAT, specifically designed for the Astro-Pi mission, allows Raspberry Pi to sense the world around it. Two ruggedized versions of Sense HAT were flown on the International Space Station from December 2015. The codes for both the versions were written by school children.

Following are some of the characteristics of the Sense HAT board –

- ❖ It has a 8X8 RGB LED matrix.
- ❖ It has a 5-button joystick.
- ❖ It also has sensors to measure acceleration, temperature, pressure, humidity, and magnetism.
- ❖ It also has sensors to measure the gyroscope.
- ❖ It has an extensive Python library associated with it. This Python library allows for easy access to this board.

The comprehensive coverage of using the Sense HAT can be found at <https://www.raspberrypi.org/learning/getting-started-with-the-sense-hat/>.



### **The Skywriter HAT**

The Skywriter HAT is an electric near-field 3D proximity interface, which senses things floating in the air over it.

Some of the characteristics of the skywriter HAT are as follows –

- You can make gestures with your hands because it can detect the motion of your hand with X, Y and Z positions.
- It can detect gestures such as flicks right, left, up, and down.
- It can easily detect the circular motion of the finger.
- It can also detect the taps directly to its surface.
- The range of the skywriter HAT has a range of about 5 cms.
- It can also be mounted behind any non-conducting surface.



The source of the above image is [www.magpi.raspberrypi.org](http://www.magpi.raspberrypi.org)

## The Xtrinsic Sense Board

The Xtrinsic Sense Board, made in partnership with the component distributors and Raspberry Pi co-manufacturer Farnell, is a low-cost sensor board. It is somewhat like the Sensor HAT, but without LEDs.

Some of the characteristics of the Xtrinsic Sense Board are as follows –

- ❖ It contains a high-precision pressure sensor. The range of this sensor is 50 to 110 kPa.
- ❖ It also contains a 3-axis digital accelerometer and a 3-D magnetometer.



## 8.12. Raspberry Pi - Third-party Software Package:

Previously, we have discussed how we can download and install the software on our Raspberry Pi. It is one of the best things about Raspberry Pi.

Here, we will be discussing some of the software packages in Raspberry Pi.

### Penguins Puzzle

This is a 3D puzzle game in which you need to safely escort a penguin to exit without letting it fall. It has a total of 50 levels. To move around, you can use the cursor keys. Shortcut key for zoom-out is Z and for reset is R.

Penguin Puzzle is preinstalled with Raspbian but in case, if you want to install or update it, you can find it on ADD/REMOVE software menu.

You can also use the following shell command to install/update it –

```
sudo apt-get install penguinspuzzle
```

Once installed, you can start playing this game by typing penguinspuzzle on shell.

## FocusWriter

FocusWriter, as the name implies, is a word processing software designed for distraction-free work. While working in FocusWriter, you will only see your writing on screen. To get the menus for changing the setting and saving your file, you need to move the mouse to the top of the screen.

One of the best things about this word processor is that you can set your daily goals for the number of words written per day or time spent for writing. To check your progress or counting number of words, you need to move the mouse to the bottom of the screen.

To install or update it, you can search it on the ADD/REMOVE software menu.

You can also use the following shell command to install/update it –

```
sudo apt-get install focuswriter
```

Once installed, to get started with FocusWriter, you need to program's entry in the office category of the Application menu.

## Mathematica

Mathematica, pre-installed on Raspbian, is a symbolic package or Computer Algebra System (CAS). In Mathematica, you can do anything with numbers, complex multidimensional graphics, and music.

As it is preinstalled, to get started with it, you need to click the Mathematica icon under the Programming category of Application menu.

You can expand equations as well as plot graphs with Mathematica.

## RealVNC

RealVNC, a remote access server and viewer software was included in Raspbian on 28th September 2016. With the help of RealVNC's new capture technology, you can directly render content. It can also be used to view non-X11 applications and to control them remotely.

## Steam Link

Steam Link which can be implemented as both hardware and software solutions, supports the streaming of the Steam content from a PC to a mobile device or other monitor.



In 2015, it was originally released as a hardware device but on 13 December 2018, its developer Valve released the official Steam Link game streaming client for Raspberry Pi microcomputer (Raspberry Pi3 and Pi 3 B+).

### **XInvaders 3D**

XInvaders 3D is a game like classic Arcade Cabinets. Similar to another classic game Asteroids, this game uses the line graphics to put a fresh spin on Space Invaders.

The three-dimensional rendering makes the aliens move progressively closer to you. To line up your shots, you need to move in four directions. The cursor keys are used to move in four directions, and you can use a spacebar to fire the shots.

To install or update it, you can search it on the ADD/REMOVE software menu.

You can also use the following shell command to install/update it –

```
sudo apt-get install xinv3d
```

Once installed, to get started with XInvaders 3D, you need to click the icon on the taskbar to go into the Terminal and then enter xinv3d.

### **Tux Paint**

Tux Paint is a simple drawing program for kids. The tools in Tux Paint help children to quickly create art on Raspberry Pi computer. It also enables freehand drawing and the placement of shapes.

The magic tool of Tux Paint can be used to create effects such as brick walls, flowers, rainbows, waves, and various other creative image distortions. It also has a stamp tool, which is used to stamp clip art onto the screen. The stamp tool includes animals, penguins, hats, food as well as musical instruments.

The name of Tux Paint is a tribute to the penguin Tux, the official mascot of the Linux kernel.

To install or update it, you can search it on the ADD/REMOVE software menu.

You can also use the following shell command to install/update it –

```
sudo apt-get install tuxpaint
```

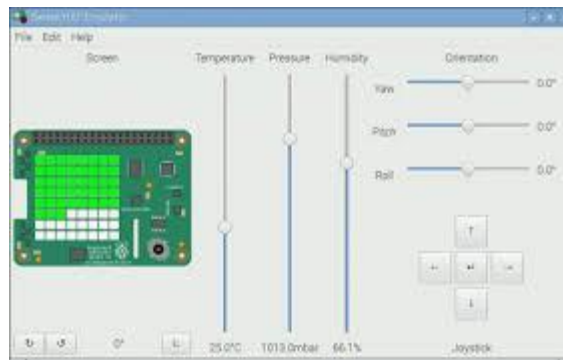
Once installed, to get started with Tux Paint, you need to click the icon on the Education category of your application menu.

### **Sense HAT Emulator**

In the previous chapter, we have discussed the Sense HAT board. Raspbian OS provides us with an emulator to use Sense HAT. You can get it in the Programming section of the Application menu.

As the name suggests, the Sense HAT emulator enables us to simulate the I/O of the Sense HAT, so that we can test how our programs work. It also provides us several sample programs to get started with.

Check the diagram of Sense HAT emulator below –



The source of the above image is [www.raspberrypi.org](http://www.raspberrypi.org)

### **Brain Party**

Brain Party, a series of fun minigames, is designed to tune up your brain between programming sessions. To get your “brain weight” score, you need to complete five randomly selected tests. The puzzles in Brain Party game will challenge your memory, logical skills, as well as observational skills.

To install or update it, you can search it on the ADD/REMOVE software menu.

You can also use the following shell command to install/update it –

```
sudo apt-get install brainparty
```

Once installed, to get started with Brain Party, you can find it under Games of the Application menu. You can also get it by typing brainparty in the command line.

## Grisbi

Grispi is a free application with the help of which you can keep track of your regular and one-off payments. It is mainly used to manage your home accounts on your Raspberry Pi computer. The format in which most of the banks enable us to download bank statements, can be easily used in Grisbi.

To install it or update it, you can search it on the ADD/REMOVE software menu.

You can also use the following shell command to install/update it –

```
sudo apt-get install grisbi
```

Once installed, to get started with Brain Party, you can find it in the Office Category of the Application menu.

### 8.13. Difference between Arduino and Raspberry Pi:

There are a wide variety of controller boards that we can use for our hardware projects. The two most popular among them are: Arduino and Raspberry Pi. Arduino is based on the ATmega family and has a relatively simple design and software structure. Raspberry Pi, basically is a single-board computer. Both of them have a CPU which executes the instructions, timers, memory and I/O pins. The key distinction between the two is that Arduino tends to have a strong I/O capability which drives external hardware directly. Whereas Raspberry Pi has a weak I/O which requires transistors to drive the hardware.

Let's see the difference between Arduino and Raspberry Pi :-

| S No. | Arduino                                                                                                                   | Raspberry Pi                                                                       |
|-------|---------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------|
| 1.    | In the year 2005, the classrooms of the Interactive Design Institute in Ivrea, Italy, first introduced the Arduino board. | In the year 2012, Eben Upton first introduced the Raspberry Pi device in February. |
| 2.    | Control unit of the Arduino is from the Atmega family.                                                                    | The control unit of Raspberry Pi is from the ARM family.                           |

| <b>S No.</b> | <b>Arduino</b>                                                                                  | <b>Raspberry Pi</b>                                                                                                                          |
|--------------|-------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| 3.           | Arduino is based on a microcontroller.                                                          | While Raspberry Pi is based on a microprocessor.                                                                                             |
| 4.           | It is designed to control the electrical components connected to the circuit board in a system. | While Raspberry Pi computes data and produces valuable outputs, and controls components in a system based on the outcome of its computation. |
| 5.           | Arduino boards have a simple hardware and software structure.                                   | While Raspberry Pi boards have a complex architecture of hardware and software.                                                              |
| 6.           | CPU architecture: 8 bit.                                                                        | CPU architecture: 64 bit.                                                                                                                    |
| 7.           | It uses very little RAM, 2 kB.                                                                  | While Raspberry Pi requires more RAM, 1 GB.                                                                                                  |
| 8.           | It clocks a processing speed of 16 MHz.                                                         | While Raspberry Pi clocks a processing speed of 1.4 GHz.                                                                                     |
| 9.           | It is cheaper in cost.                                                                          | While Raspberry Pi is expensive.                                                                                                             |
| 10.          | It has a higher I/O current drive strength.                                                     | While Raspberry Pi has a lower I/O current drive strength.                                                                                   |
| 11.          | It consumes about 200 MW of power.                                                              | While it consumes about 700 MW of power.                                                                                                     |
| 12.          | Its logic level is 5V.                                                                          | Its logic level is 3V.                                                                                                                       |
| 13.          | It does not have internet support.                                                              | It has inbuilt Ethernet port and WiFi support.                                                                                               |
| 14.          | It has higher current drive strength.                                                           | It has lower current drive strength.                                                                                                         |

| <b>S No.</b> | <b>Arduino</b>                                                                                   | <b>Raspberry Pi</b>                                                                                  |
|--------------|--------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------|
| 15.          | Some of the applications of Arduino are traffic light countdown timer , Weighing machines , etc. | Some of the applications of Raspberry Pi are Stop motion cameras , Robot Controllers , Game Servers. |
| 16.          | Operating systems are required in Arduino.                                                       | Operating System is required in Raspberry Pi.                                                        |
| 17.          | Two tiny cores Arduino with 32 Mhz                                                               | Single core and 700 MHz                                                                              |

## 8.14. Raspberry Pi Projects:

### Project – 01

#### **LED Blinking with Raspberry Pi and Python Program**

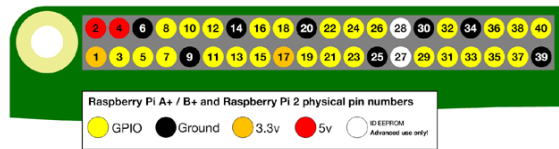
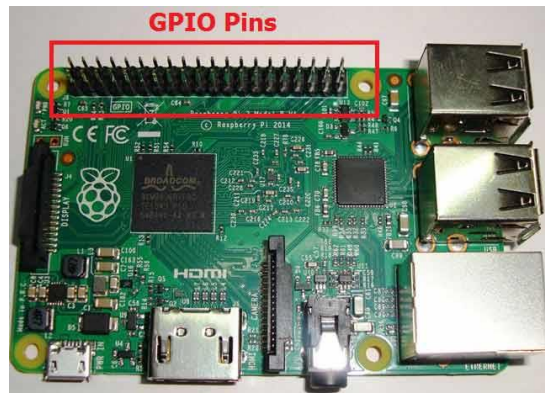
---

Raspberry Pi is a pocket sized computer which also have GPIO pins for connecting it to other sensors and peripherals which makes it a good platform for embedded engineers. It has an ARM architecture processor based board designed for electronic engineers and hobbyists. The PI is one of most trusted project development platforms out there now. With higher processor speed and high RAM, the Raspberry Pi can be used for many high profile projects like Image processing and Internet of Things. Raspberry Pi 4 with 8GB RAM is the high end version available for sale now. It also has other lower version with 4GB and 2GB RAM.

For doing any of high profile projects, one need to understand the basic functions of PI. That is why we are here, we will be teaching all the basic functionalities of Raspberry Pi in these tutorials. In each tutorial series we will discuss one of functions of PI. By the end of tutorial series you will be able to do high profile projects by yourself. Check these for Getting Started with Raspberry Pi and Raspberry Pi Configuration.

Here, we will understand the concept of writing and executing programs on PYTHON. We will start with Blink LED using Raspberry Pi. Raspberry Pi LED Blink is done by connecting an LED to one of GPIO pins of PI and turning it ON and OFF. After learning the basics of Raspberry Pi, you can move on its high end applications, which we have covered in our dedicated Raspberry Pi section and can also check basics by following interfacing a button with Raspberry Pi, Raspberry Pi PWM tutorial, using DC motor with Raspberry Pi etc.

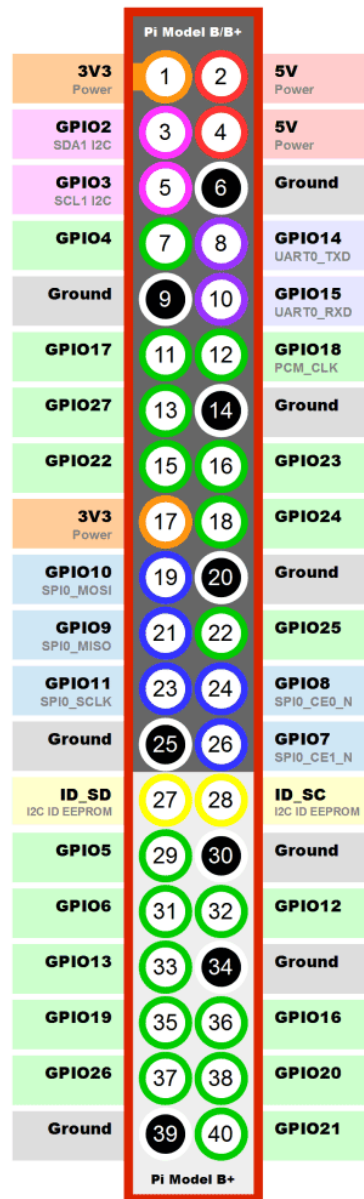
We will discuss a bit about PI GPIO Pins before going any further,



As shown in above figure, there are 40 output pins for the PI. But when you look at the second figure, you can see not all 40 pin out can be programmed to our use. These are only 26 GPIO pins which can be programmed. These pins go from GPIO2 to GPIO27.

These 26 GPIO pins can be programmed as per need. Some of these pins also perform some special functions, we will discuss about that later. With special GPIO put aside, we have 17 GPIO remaining (Light green Circle).

Each of these 17 GPIO pins can deliver a maximum of 15mA current. And the sum of currents from all GPIO cannot exceed 50mA. So we can draw a maximum of 3mA in average from each of these GPIO pins. So one should not tamper with these things unless you know what you are doing.



## Components Required

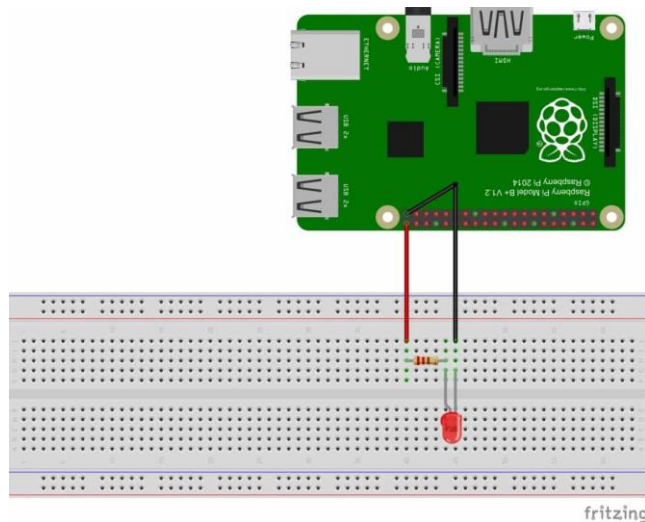
Here we are using Raspberry Pi 2 Model B with Raspbian Jessie OS. All the basic Hardware and Software requirements are previously discussed, you can look it up in the Raspberry Pi Introduction, other than that we need:

- ❖ Connecting pins
- ❖ 220Ω or 1KΩ resistor
- ❖ LED
- ❖ Bread Board



## Circuit Explanation:

Circuit diagram for Raspberry Pi LED Blink is given below:

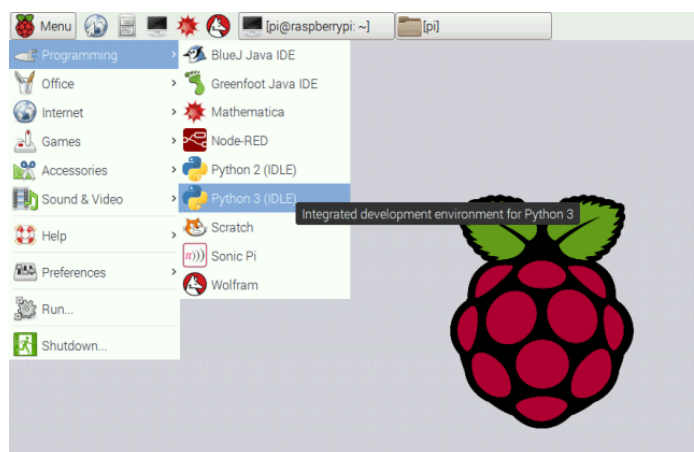


As shown in the circuit diagram we are going to connect an LED between PIN40 (GPIO21) and PIN39 (GROUND). As said earlier, we cannot draw more than 15mA from any one of these pins, so to limit the current we are connecting a 220 $\Omega$  or 1K $\Omega$  resistor in series with the LED.

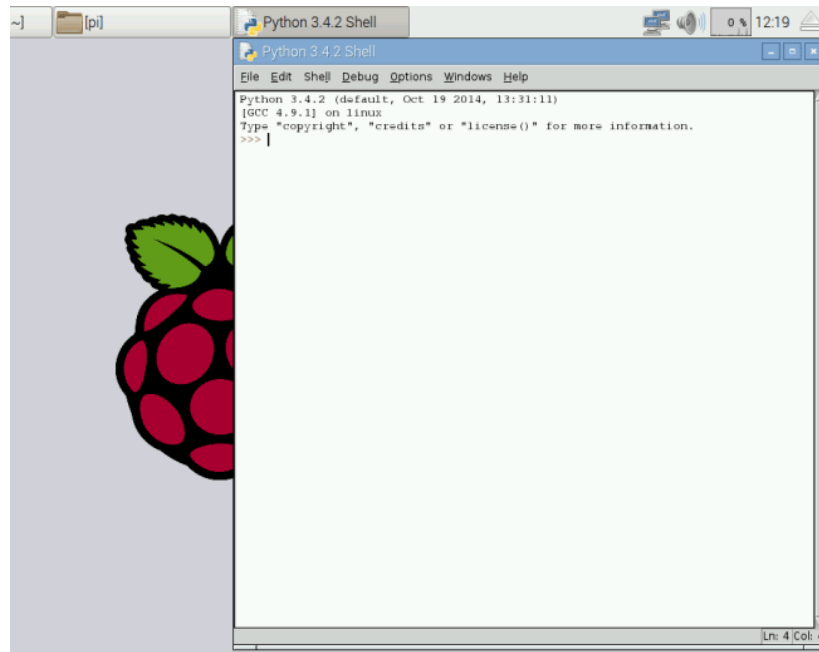
## Working Explanation:

Since we have everything ready, turn ON your PI and go to the desktop.

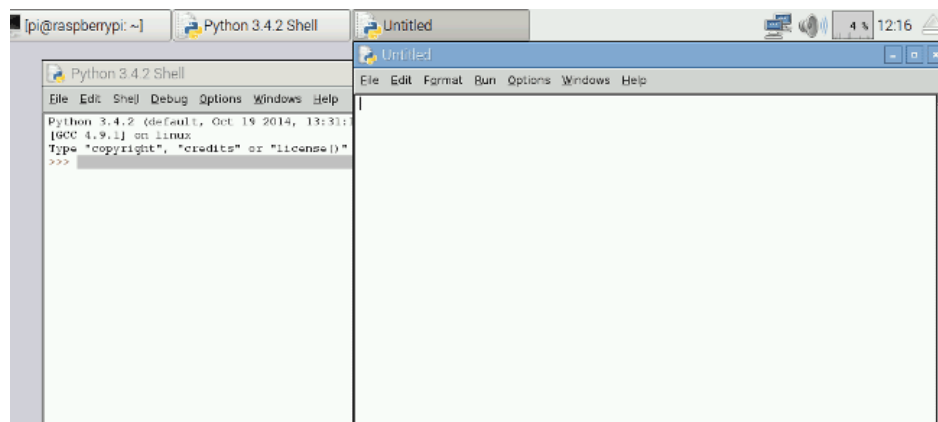
1. On the desktop, go the Start Menu and choose for the PYTHON 3, as shown in figure below.



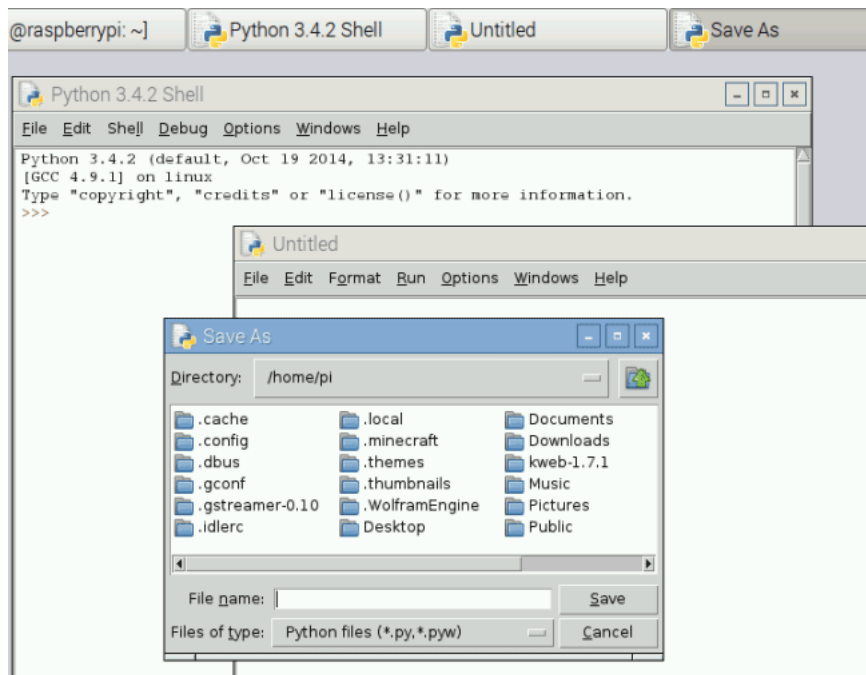
2. After that, PYTHON will run and you will see a window as shown in below figure.



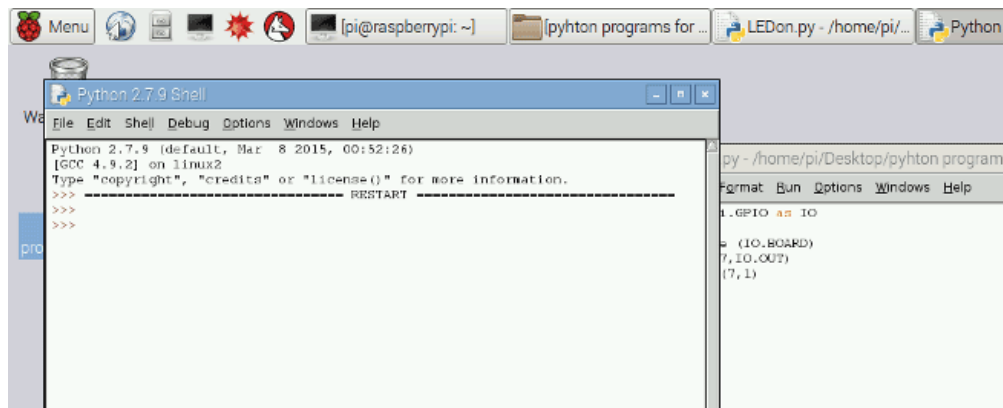
3. After that, click on New File in File Menu, You will see a new Window open,



4. Save this file as blinky on the desktop,



5. After that write the program for blinky as given below and execute the program by clicking on “RUN” on ‘DEBUG’ option.

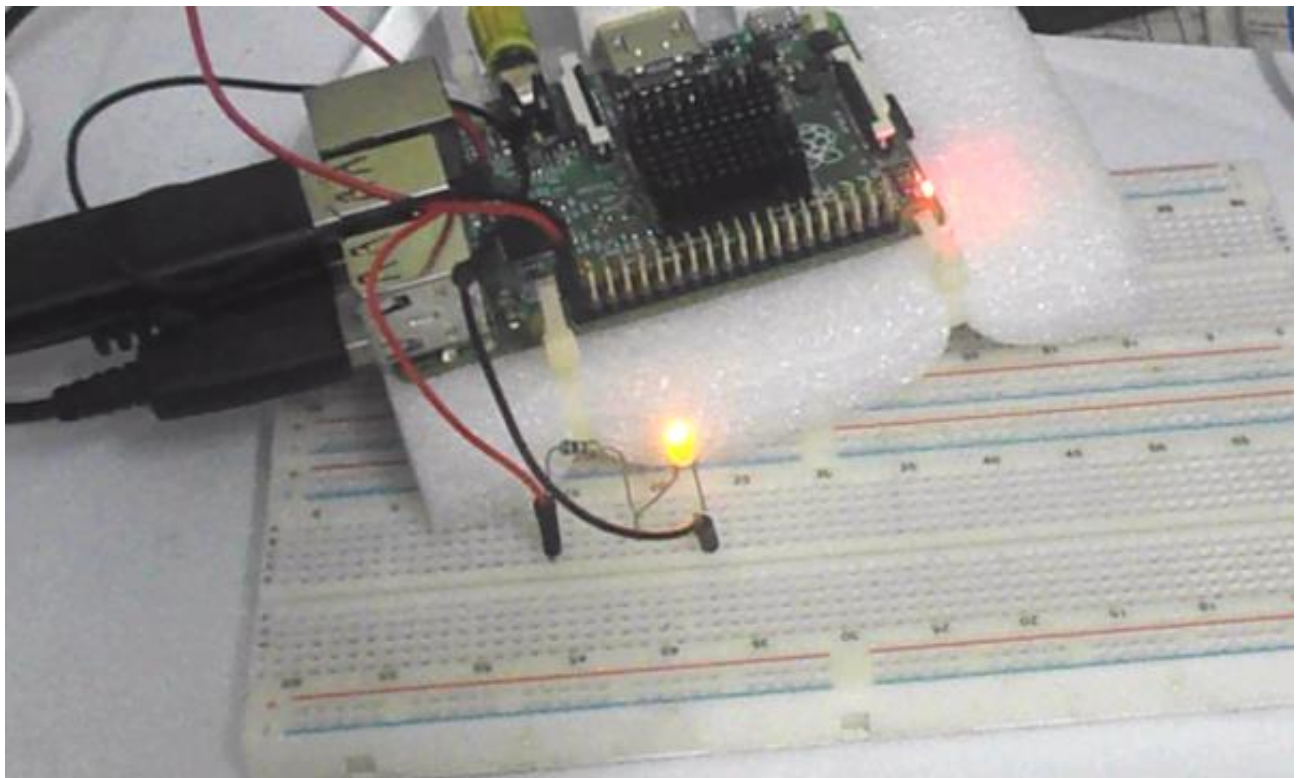


If the program has no errors in it, you will see a “>>>”, which means the program is executed successfully. By this time you should see the LED blinking three times. If there were any errors in the program, the execution tells to correct it. Once the error is corrected execute the program again.

Complete PYTHON program Code for LED Blinking is given below.

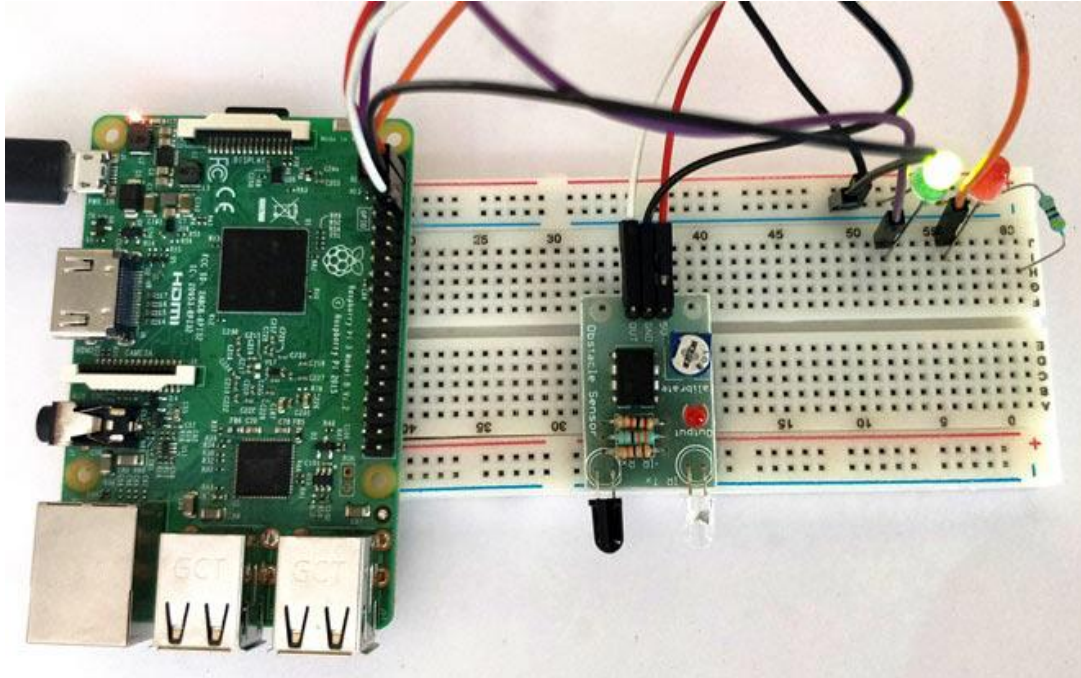
## Code:

```
import RPi.GPIO as IO # calling header file for GPIO's of PI
import time # calling for time to provide delays in program
IO.setmode (IO.BOARD) # programming the GPIO by BOARD pin numbers, GPIO21 is cal
l e d a s P I N 4 0
IO.setup(40,IO.OUT) # initialize digital pin40 as an output.
IO.output(40,1) # turn the LED on (making the voltage level HIGH)
time.sleep(1) # sleep for a second
IO.cleanup() # turn the LED off (making all the output pins LOW)
time.sleep(1) #sleep for a second
#loop is executed second time
IO.setmode (IO.BOARD)
IO.setup(40,IO.OUT)
IO.output(40,1)
time.sleep(1)
IO.cleanup()
time.sleep(1)
#loop is executed third time
IO.setmode (IO.BOARD)
IO.setup(40,IO.OUT)
IO.output(40,1)
time.sleep(1)
IO.cleanup()
time.sleep(1)
```



## Project-02

### Interfacing IR Sensor to Raspberry Pi



As we all know Raspberry Pi is a wonderful Developing platform based on ARM microprocessor. With its high computational power it can work out wonders in hands of electronics hobbyists or students. All this can be possible only if we know how to make it interact with the real world. There are many sensors which can detect certain parameters from the real time world and transfer it to a digital world. We have covered lot of Raspberry Pi Projects with many sensors. Raspberry Pi is also a boon for IoT projects, as it is a pocket sized computer with inbuilt Wi-Fi, having capabilities of a microcontroller.

Here, we will learn how we can Interface an IR sensor with Raspberry pi. These sensors are most commonly use in small robots like line follower robot, Edge avoiding robot etc.. Simply putting, it can detect the presence of objects before it and also differentiate between white and black colour. Sounds cool right?

So lets learn how to interface this sensor with Raspberry Pi. In this project, when there is no object in front of IR sensor then the Red LED remains turned on and soon as we put

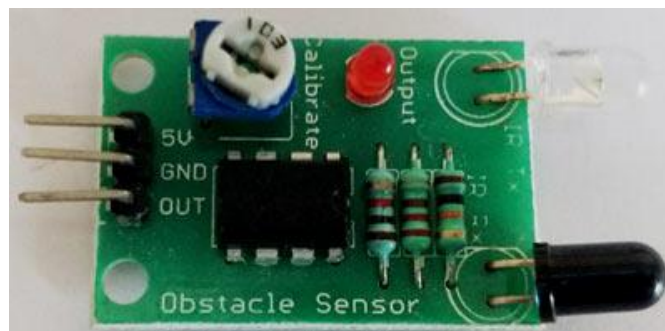
something in front of IR sensor then red LED turns off and Green LED turn on. This circuit can also serve as Security Alarm Circuit.

#### Material Required:

1. Raspberry Pi 3 (any model)
2. IR sensor Module
3. Green and Red LED lights
4. Breadboard
5. Connecting wires

#### IR Sensor Module:

IR sensors (Infrared sensor) are modules which detect the presence of objects before them. If the object is present it give 3.3V as output and if it is not present it gives 0 volt. This is made possible by using a pair of IR pair (transmitter and receiver), the transmitter (IR LED) will emit an IR ray which will get reflected if there is a object present before it. This IR ray will be received back by the receiver (Photodiode) and the output will be made high after amplified using an op-amp link LM358. You can learn more about IR Sensor Module Circuit [here](#).



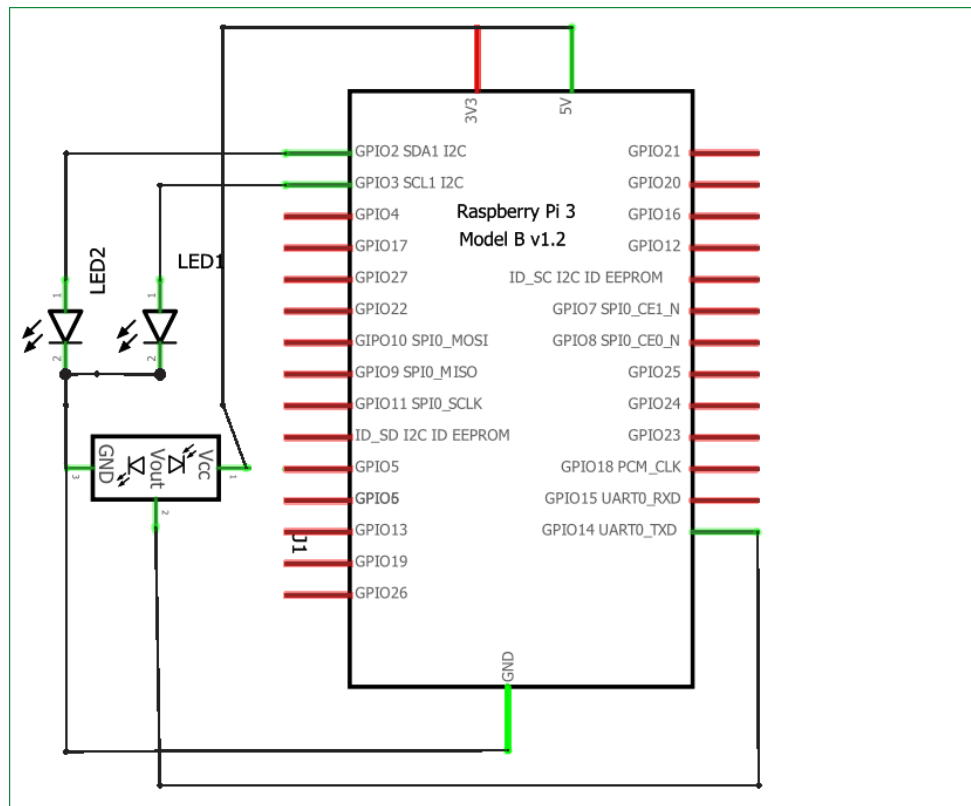
The IR Sensor used in this project is shown above. Like all IR sensor it has three pins which are 5V, Gnd and Out respectively. The module is powered by the 5V pin from Raspberry Pi and the out pin is connected to GPIO14 of Raspberry Pi. The potentiometer on top of the module can be used to adjust the range of the IR sensor.

#### Circuit Diagram and Explanation:

The circuit diagram for connecting Raspberry Pi with IR sensor is shown below. As you can see the circuit diagram is very simple. We have directly powered the IR module from the 5V



and Ground Pin of Raspberry Pi. The output pin of the IR module is connected to the GPIO14. We have also used two LED (Green and Red) to indicate the status of the object. These two LEDs are connected to GPIO3 and GPIO2 respectively.



Since the GPIO pins of Raspberry Pi are 3.3V, a current limiting resistor is not mandatory. However if desired a resistor of value 470 ohms can be added between the ground pin of LEDs and Raspberry Pi. The whole circuit is powered by a 5V mobile charger through the micro USB port of the Raspberry pi.

**Note:** When connecting any sensor, make sure the ground of the sensor is connected to ground of the MCU or MPU (here Raspberry Pi). Only then they will be able to communicate.

### Programming your Raspberry Pi:

Here we are using Python Programming language for programming RPi. There are many ways to program your Raspberry Pi. In this tutorial we are using the Python 3 IDE, since it is the most used one. The complete Python program is given at the end of this tutorial. Learn more about Program and run code in Raspberry Pi here.



We will talk about few commands which we are going to use in PYTHON program,

We are going to import GPIO file from library, below function enables us to program GPIO pins of PI. We are also renaming “GPIO” to “IO”, so in the program whenever we want to refer to GPIO pins we will use the word ‘IO’.

```
import RPi.GPIO as IO
```

Sometimes, when the GPIO pins, which we are trying to use, might be doing some other functions. In that case, we will receive warnings while executing the program. Below command tells the PI to ignore the warnings and proceed with the program.

```
IO.setwarnings(False)
```

We can refer the GPIO pins of PI, either by pin number on board or by their function number. Like ‘PIN 29’ on the board is ‘GPIO5’. So we tell here either we are going to represent the pin here by ‘29’ or ‘5’.

```
IO.setmode (IO.BCM)
```

We are setting 3 pins as input/output pins. The two output pins will control the LED and the input pin will read signal from the IR sensor.

```
IO.setup(2,IO.OUT) #GPIO 2 -> Red LED as output
IO.setup(3,IO.OUT) #GPIO 3 -> Green LED as output
IO.setup(14,IO.IN) #GPIO 14 -> IR sensor as input
```

Now we have to turn off the Green LED and turn on the Red LED when the object is far. This can be done by checking the GPIO14 pin.

```
if(IO.input(14)==True): #object is far away
 IO.output(2,True) #Red led ON
 IO.output(3,False) # Green led OFF
```

Similarly we have to turn on the Green LED and turn off the Red LED when the object is near.

```
if(IO.input(14)==False): #object is near

 IO.output(3,True) #Green led ON

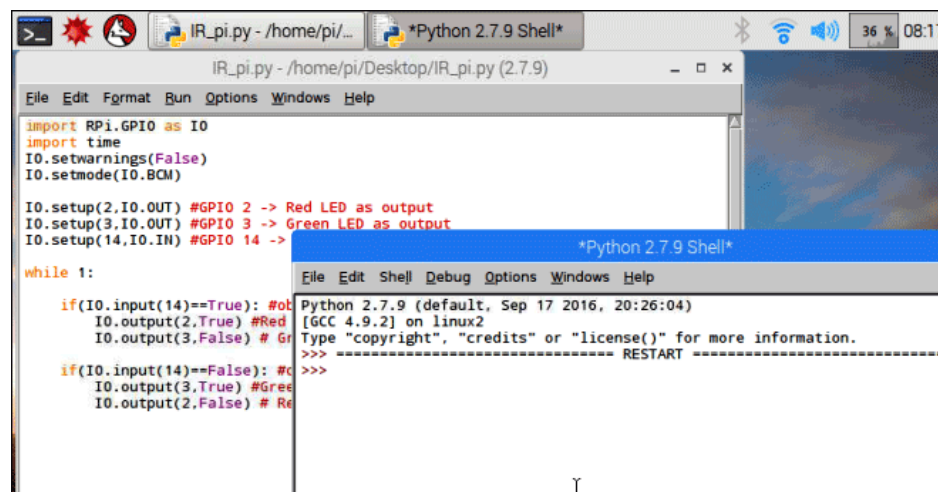
 IO.output(2,False) # Red led OFF
```

Below command is used as forever loop, with this command the statements inside this loop will be executed continuously.

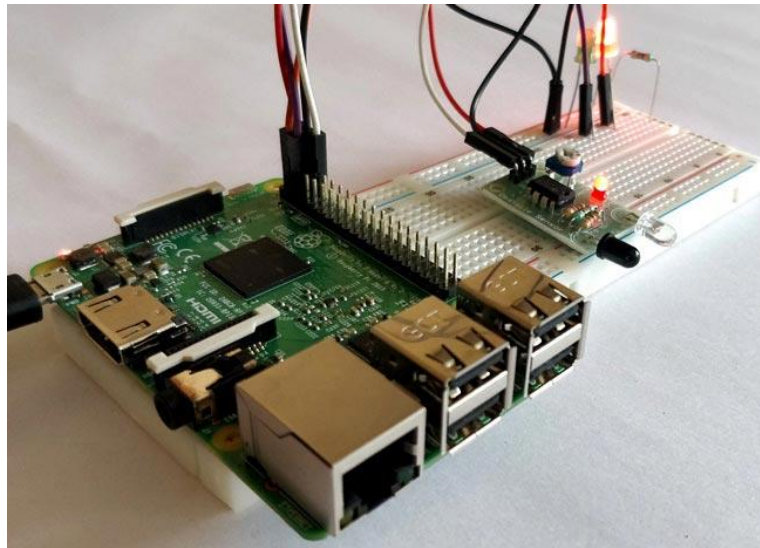
While 1:

## Working:

Once you have created your python code, execute it using the run command. If the program is executed without any errors you should get the following screen.



You should also see the red colour LED going high when there is no object in front of the sensor as shown below.



Now, bring something close to the IR led and you should notice the red LED turning off and the Green turning on. Complete working can be found on the Video given below.

Hope you understood the project and were able to build something useful with it. If any queries post those on the comment section below or on the forum.

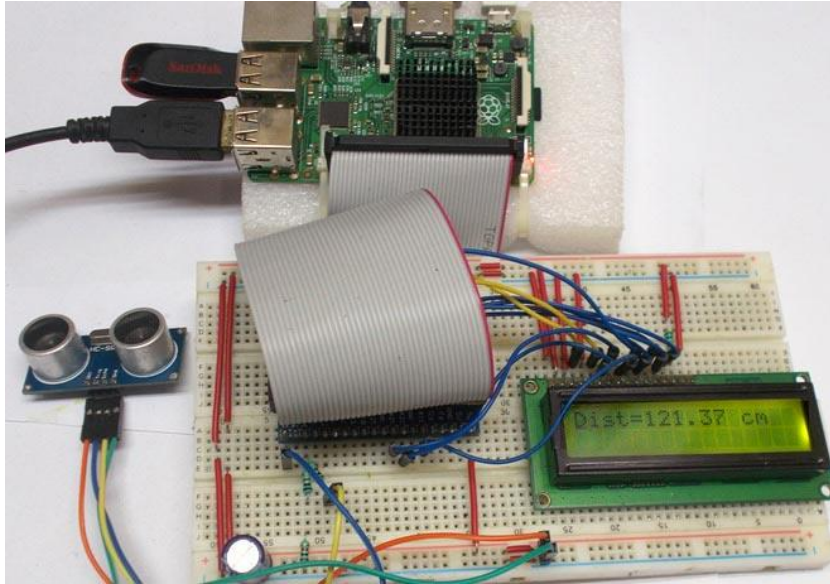
## Code

```
import RPi.GPIO as IO
import time
IO.setwarnings(False)
IO.setmode(IO.BCM)
IO.setup(2,IO.OUT) #GPIO 2 -> Red LED as output
IO.setup(3,IO.OUT) #GPIO 3 -> Green LED as output
IO.setup(14,IO.IN) #GPIO 14 -> IR sensor as input
while 1:
 if(IO.input(14)==True): #object is far away
 IO.output(2,True) #Red led ON
 IO.output(3,False) # Green led OFF
 if(IO.input(14)==False): #object is near
 IO.output(3,True) #Green led ON
 IO.output(2,False) # Red led OFF
```

### Project – 03

## Measuring Distance using Raspberry Pi and HCSR04 Ultrasonic Sensor

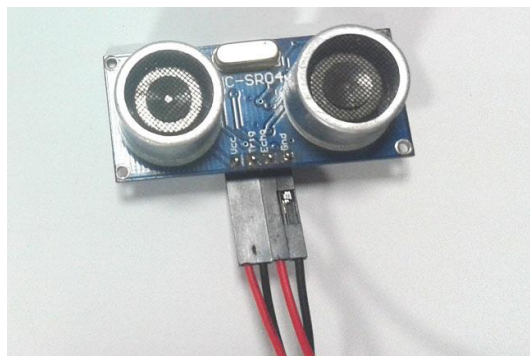
---



In this project we are going to interface HC-SR04 Ultrasonic sensor module to Raspberry Pi to measure distance. We have previously used Ultrasonic sensor with Raspberry Pi to build Obstacle Avoiding Robot. Before going any further, let's know about Ultrasonic sensor.

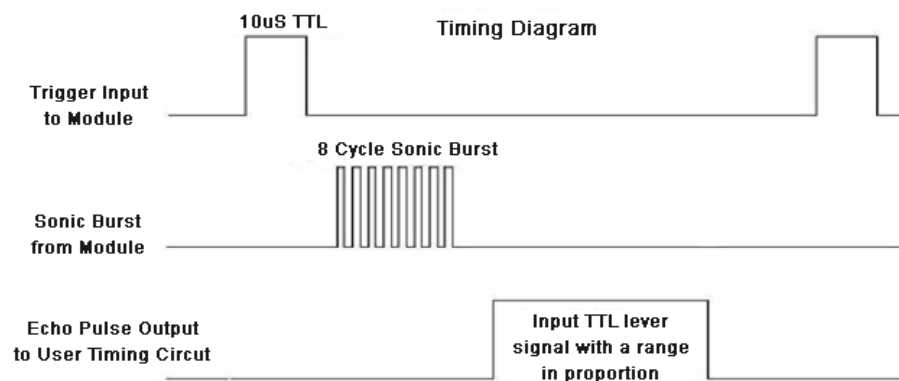
### **HC-SR04 Ultrasonic Sensor:**

The Ultrasonic Sensor is used to measure the distance with high accuracy and stable readings. It can measure distance from 2cm to 400cm or from 1 inch to 13 feet. It emits an ultrasound wave at the frequency of 40KHz in the air and if the object will come in its way then it will bounce back to the sensor. By using that time which it takes to strike the object and comes back, you can calculate the distance.



The ultrasonic sensor uses a technique called “ECHO”. “ECHO” is simply a reflected sound wave. You will have an ECHO when sound reflects back after reaching a dead end.

HCSR04 module generates a sound vibration in ultrasonic range when we make the ‘Trigger’ pin high for about 10us which will send a 8 cycle sonic burst at the speed of sound and after striking the object, it will be received by the Echo pin. Depending on time taken by sound vibration to get back, it provides appropriate pulse output. If the object is far away then it takes more time for ECHO to be heard and the output pulse width will be big. And if the obstacle is near, then the ECHO will be heard faster and output pulse width will be smaller.



We can calculate the distance of the object based on the time taken by ultrasonic wave to return back to the sensor. Since the time and speed of sound is known we can calculate the distance by the following formulae.

✓ Distance= (Time x Speed of Sound in Air (343 m/s))/2.

The value is divided by two since the wave travels forward and backward covering the same distance. Thus the time to reach obstacle is just half the total time taken

So Distance in centimeter = 17150\*T

### Components Required:

Here we are using Raspberry Pi 2 Model B with Raspbian Jessie OS. All the basic Hardware and Software requirements are previously discussed, you can look it up in the Raspberry Pi Introduction and Raspberry PI LED Blinking for getting started, other than that we need:

- ❖ Raspberry Pi with pre-installed OS
- ❖ HC-SR04 Ultrasonic Sensor
- ❖ Power supply (5v)
- ❖ 1K $\Omega$  resistor (3 pieces)
- ❖ 1000uF capacitor
- ❖ 16\*2 character LCD

### Circuit Explanation:

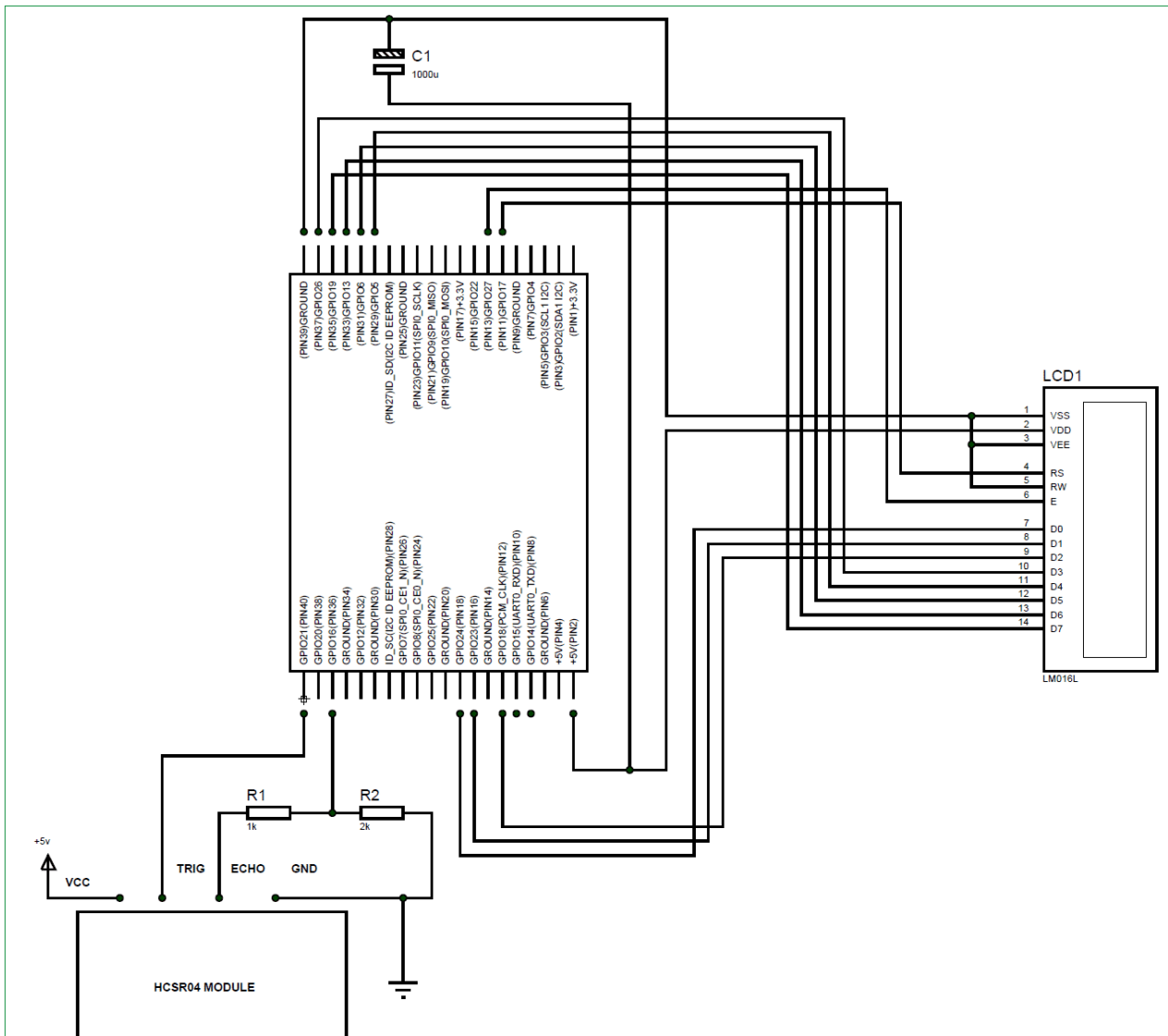
Connections between Raspberry Pi and LCD are given in the below table:

| LCD connection | Raspberry Pi connection |
|----------------|-------------------------|
| GND            | GND                     |
| VCC            | +5V                     |
| VEE            | GND                     |
| RS             | GPIO17                  |
| R/W            | GND                     |
| EN             | GPIO27                  |
| D0             | GPIO24                  |
| D1             | GPIO23                  |
| D2             | GPIO18                  |
| D3             | GPIO26                  |

|    |        |
|----|--------|
| D4 | GPIO5  |
| D5 | GPIO6  |
| D6 | GPIO13 |
| D7 | GPIO19 |

In this circuit, we used 8bit communication (D0-D7) to connect LCD with Raspberry Pi, however this is not a compulsory, we can also use 4-bit communication (D4-D7), but with 4 bit communication program becomes a bit complex for beginners so just go with 8 bit communication. Here we have connected 10 pins of LCD to Raspberry Pi in which 8 pins are data pins and 2 pins are control Pins.

Below is the circuit diagram for connecting HC-SR04 sensor and LCD with Raspberry Pi for measuring the distance.



As shown in the figure, HC-SR04 Ultrasonic Sensor has four pins,

1. PIN1- VCC or +5V
2. PIN2- TRIGGER (10us High pulse given to tell the sensor to sense the distance)
3. PIN3- ECHO (Provides pulse output whose width represents distance after trigger)
4. PIN4- GROUND

Echo pin provides +5V output pulse which cannot be connected to Raspberry Pi directly. So we will be using Voltage Divider Circuit (built using R1 and R2) to get +3.3V logic instead of +5V logic.



## Working Explanation:

Complete working of Raspberry Pi Distance Measure goes as,

1. Triggering the sensor by pulling up the trigger pin for 10uS.
2. Sound wave is sent by the sensor. After receiving the ECHO, sensor module provides an output proportional to distance.
3. We will record the time when the output pulse goes from LOW to HIGH and when again when its goes form HIGH to LOW.
4. We will have start and stop time. We will use distance equation to calculate the distance.
5. The distance is displayed in 16x2 LCD display.

Accordingly we have written the Python Program for Raspberry Pi to do the following functions:

1. To send trigger to sensor
2. Record start and stop time of pulse output from sensor.
3. To Calculate the distance by using START and STOP time.
4. To Display the result obtained on the 16\*2 LCD.

Complete Program is given below. Program is well explained through the comments, if you have any doubt you can ask in comment section below.

### Code:

```
import time
import RPi.GPIO as IO #calling for header file which helps in using GPIOs of PI
string_of_characters = 0
IO.setwarnings(False) #do not show any warnings
IO.setmode (IO.BCM) #programming the GPIO by BCM pin numbers. (like PIN29 as GPIO5
)
```

```

IO.setup(17,IO.OUT) #initialize GPIO17,27,24,23,18,26,5,6,13,19,21 as an output
IO.setup(27,IO.OUT)
IO.setup(24,IO.OUT)
IO.setup(23,IO.OUT)
IO.setup(18,IO.OUT)
IO.setup(26,IO.OUT)
IO.setup(5,IO.OUT)
IO.setup(6,IO.OUT)
IO.setup(13,IO.OUT)
IO.setup(19,IO.OUT)
IO.setup(21,IO.OUT)
IO.setup(16,IO.IN) #initialize GPIO16 as an input
def send_a_command (command): #steps for sending a command to 16x2 LCD
 pin=command
 PORT(pin);
 IO.output(17,0)
 #PORTD&= ~(1<<RS);
 IO.output(27,1)
 #PORTD|= (1<<E);
 time.sleep(0.001)
 #_delay_ms(50);
 IO.output(27,0)
 #PORTD&= ~(1<<E);
 pin=0
 PORT(pin);
def send_a_character (character): #steps for sending a character to 16x2 LCD
 pin=character
 PORT(pin);
 IO.output(17,1)
 #PORTD|= (1<<RS);
 IO.output(27,1)
 #PORTD|= (1<<E);

```

```

time.sleep(0.001)
#_delay_ms(50);
IO.output(27,0)
#PORTD&= ~(1<<E);
pin=0
PORT(pin);
def PORT(pin): #assigning level for PI GPIO for sending data to LCD through D0-D
7
 if(pin&0x01 == 0x01):
 IO.output(24,1)
 else:
 IO.output(24,0)
 if(pin&0x02 == 0x02):
 IO.output(23,1)
 else:
 IO.output(23,0)
 if(pin&0x04 == 0x04):
 IO.output(18,1)
 else:
 IO.output(18,0)
 if(pin&0x08 == 0x08):
 IO.output(26,1)
 else:
 IO.output(26,0)
 if(pin&0x10 == 0x10):
 IO.output(5,1)
 else:
 IO.output(5,0)
 if(pin&0x20 == 0x20):
 IO.output(6,1)
 else:
 IO.output(6,0)

```

```

if(pin&0x40 == 0x40):
 IO.output(13,1)
else:
 IO.output(13,0)
if(pin&0x80 == 0x80):
 IO.output(19,1)
else:
 IO.output(19,0)
def send_a_string(string_of_characters):
 string_of_characters = string_of_characters.ljust(16," ")
 for i in range(16):
 send_a_character(ord(string_of_characters[i])) #send characters one by one through data port
 while 1:
 send_a_command(0x38); #16x2 line LCD
 send_a_command(0x0E); #screen and cursor ON
 send_a_command(0x01); #clear screen
 time.sleep(0.1) #sleep for 100msec
 IO.setup(21,1)
 time.sleep(0.00001)
 IO.setup(21,0) #sending trigger pulse for sensor to measure the distance
 while (IO.input(16)==0):
 start = time.time() #store the start time of pulse output
 while (IO.input(16)==1):
 stop = time.time() #store the stop time
 distance = ((stop - start)*17150) #calculate distance from time
 distance = round(distance,2) #round up the decimal values
 if(distance<400): #if distance is less than 400 cm, display the result on LCD
 send_a_command(0x80 + 0);
 send_a_string ("Dist=%s cm"% (distance));
 time.sleep(0.15)
 if(distance>400): #If distance is more than 400cm, just print 400+ on LCD
 send_a_command(0x80 + 0);

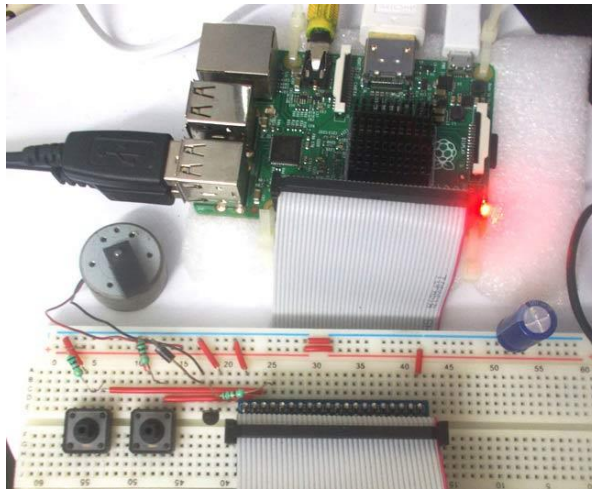
```

```
send_a_string ("Dist= 400+ cm");
time.sleep(0.15)
```

#### **Project – 04**

### **DC Motor Control with Raspberry Pi**

---

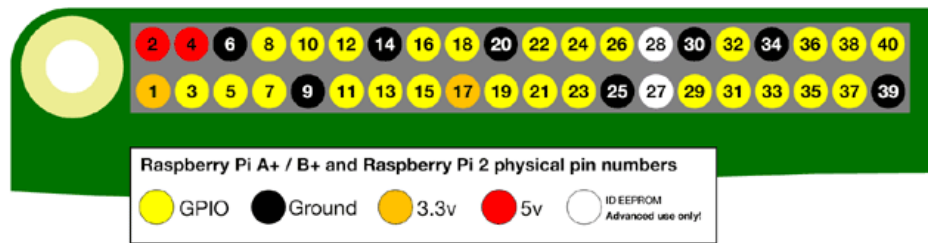


Raspberry Pi is an ARM architecture processor based board designed for electronic engineers and hobbyists. The PI is one of most trusted project development platforms out there now. With higher processor speed and 1 GB RAM, the PI can be used for many high profile projects like Image processing and Internet of Things.

For doing any of high profile projects, one need to understand the basic functions of PI. We will be covering all the basic functionalities of Raspberry Pi in these tutorials. In each tutorial we will discuss one of functions of PI. By the end of tutorial series you will be able to do high profile projects by yourself. Check these for Getting Started with Raspberry Pi and Raspberry Pi Configuration.

We have discussed LED Blinky, Button Interfacing and PWM generation in previous tutorials. In this tutorial we will Control the Speed of a DC motor using Raspberry Pi and PWM technique. PWM (Pulse Width Modulation) is a method used for getting variable voltage out of constant power source. We have discussed about PWM in the previous tutorial.

There are 40 GPIO output pins in Raspberry Pi 2. But out of 40, only 26 GPIO pins (GPIO2 to GPIO27) can be programmed. Some of these pins perform some special functions. With special GPIO put aside, we have 17 GPIO remaining. To know more about GPIO pins, go through: LED Blinking with Raspberry Pi



Each of these 17 GPIO pin can deliver a maximum of 15mA. And the sum of currents from all GPIO Pins cannot exceed 50mA. So we can draw a maximum of 3mA in average from each of these GPIO pins. So one should not tamper with these things unless you know what you are doing.

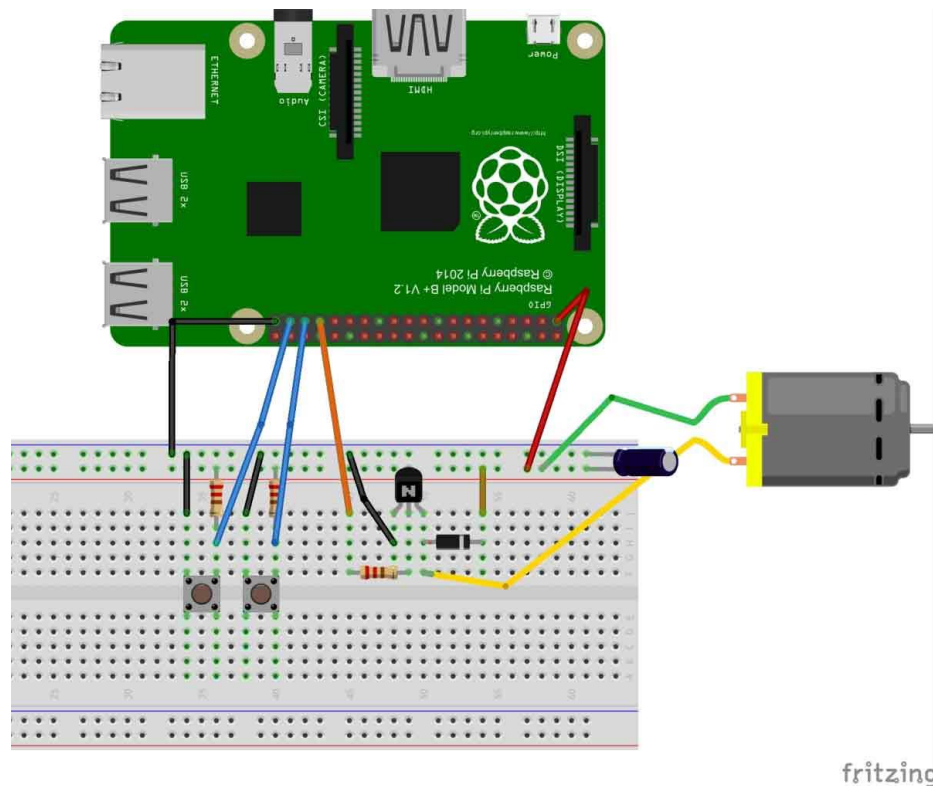
There are +5V (Pin 2 & 4) and +3.3V (Pin 1 & 17) power output pins on the board for connecting other modules and sensors. This power rail is connected in parallel to processor power. So drawing High current from this power rail affects the Processor. There is a fuse on the PI board which will trip once you apply high load. You can draw 100mA safely from the +3.3V rail. We are talking about this here because; we are connecting the DC motor to +3.3V. With the power limit in mind, we can only connect low power motor here, if you want to drive high power motor, consider powering it from a separate power source.

## Components Required:

Here we are using Raspberry Pi 2 Model B with Raspbian Jessie OS. All the basic Hardware and Software requirements are previously discussed, you can look it up in the Raspberry Pi Introduction, other than that we need:

- ❖ Connecting pins
- ❖ 220Ω or 1KΩ resistor (3)
- ❖ Small DC Motor
- ❖ Buttons (2)
- ❖ 2N2222 Transistor
- ❖ 1N4007 Diode
- ❖ Capacitor- 1000uF
- ❖ Bread Board

## Circuit Explanation:



As said earlier, we cannot draw more than 15mA from any GPIO pins and DC motor draws more than 15mA, so the PWM generated by Raspberry Pi cannot be fed to the DC motor directly. So if we connect the motor directly to PI for speed control, the board might get damaged permanently.

So we are going to use an NPN transistor (2N2222) as a switching device. This transistor here drives the high power DC motor by taking PWM signal from PI. Here one should pay attention that wrongly connecting the transistor might load the board heavily.

The motor is an induction and so while switching the motor, we experience inductive spiking. This spiking will heat up the transistor heavily, so we will be using Diode (1N4007) to provide protection to transistor against Inductive Spiking.

In order to reduce the voltage fluctuations, we will be connecting a 1000uF capacitor across the power supply as shown in the Circuit Diagram.



## Working Explanation:

Once everything is connected as per the circuit diagram, we can turn ON the PI to write the program in PYTHON.

We will talk about few commands which we are going to use in PYTHON program.

We are going to import GPIO file from library, below function enables us to program GPIO pins of PI. We are also renaming “GPIO” to “IO”, so in the program whenever we want to refer to GPIO pins we will use the word ‘IO’.

```
import RPi.GPIO as IO
```

Sometimes, when the GPIO pins, which we are trying to use, might be doing some other functions. In that case, we will receive warnings while executing the program. Below command tells the PI to ignore the warnings and proceed with the program.

```
IO.setwarnings(False)
```

We can refer the GPIO pins of PI, either by pin number on board or by their function number. Like ‘PIN 35’ on the board is ‘GPIO19’. So we tell here either we are going to represent the pin here by ‘35’ or ‘19’.

```
IO.setmode (IO.BCM)
```

We are setting GPIO19 (or PIN35) as output pin. We will get PWM output from this pin.

```
IO.setup(19,IO.OUT)
```

After setting the pin as output we need to setup the pin as PWM output pin,

```
p = IO.PWM(output channel , frequency of PWM signal)
```

The above command is for setting up the channel and also for setting up the frequency of the PWM signal. ‘p’ here is a variable it can be anything. We are using GPIO19 as the

PWM output channel. 'frequency of PWM signal' has been chosen 100, as we don't want to see LED blinking.

Below command is used to start PWM signal generation, 'DUTYCYCLE' is for setting the Turn On ratio, 0 means LED will be ON for 0% of time, 30 means LED will be ON for 30% of the time and 100 means completely ON.

```
p.start(DUTYCYCLE)
```

In case the Condition in the braces is true, the statements inside the loop will be executed once. So if the GPIO pin 26 goes low, then the statements inside the IF loop will be executed once. If the GPIO pin 26 does not goes low, then the statements inside the IF loop will not be executed.

```
if(IO.input(26) == False):
```

While 1: is used for infinity loop. With this command the statements inside this loop will be executed continuously.

We have all the commands needed to achieve the speed control with this.

After writing the program and executing it, all there is left is operating the control. We have two buttons connected to PI; one for incrementing the Duty Cycle of PWM signal and other for decrementing the Duty Cycle of PWM signal. By pressing one button the, speed of DC motor increases and by pressing the other button, the speed of DC motor decreases. With this we have achieved the DC Motor Speed Control by Raspberry Pi.

## Code

```
import RPi.GPIO as IO # calling header file which helps us use GPIO's of PI
import time # calling time to provide delays in program
IO.setwarnings(False) #do not show any warnings
x=0 #integer for storing the duty cycle value
IO.setmode (IO.BCM) #we are programming the GPIO by BCM pin numbers. (PIN35 as '
GPIO19')
```

```

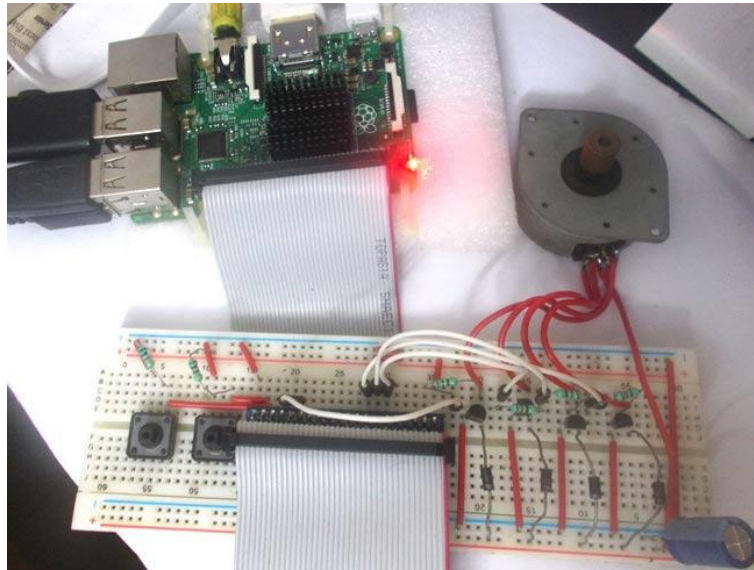
IO.setup(13,IO.OUT) # initialize GPIO13 as an output.
IO.setup(19,IO.IN) # initialize GPIO19 as an input.
IO.setup(26,IO.IN) # initialize GPIO26 as an input.
p = IO.PWM(13,100) #GPIO13 as PWM output, with 100Hz frequency
p.start(0) #generate PWM signal with 0% duty cycle
while 1: #execute loop forever
 p.ChangeDutyCycle(x) #change duty cycle for changing the brightness of LED.
 if(IO.input(26) == False): #if button1 is pressed
 if(x<50):
 x=x+1 #increment x by one if x<50
 time.sleep(0.2) #sleep for 200ms
 if(IO.input(19) == False): #if button2 is pressed
 if(x>0):
 x=x-1 #decrement x by one if x>0
 time.sleep(0.2) #sleep for 200ms

```

## Project – 05

### Stepper Motor Control with Raspberry Pi

---



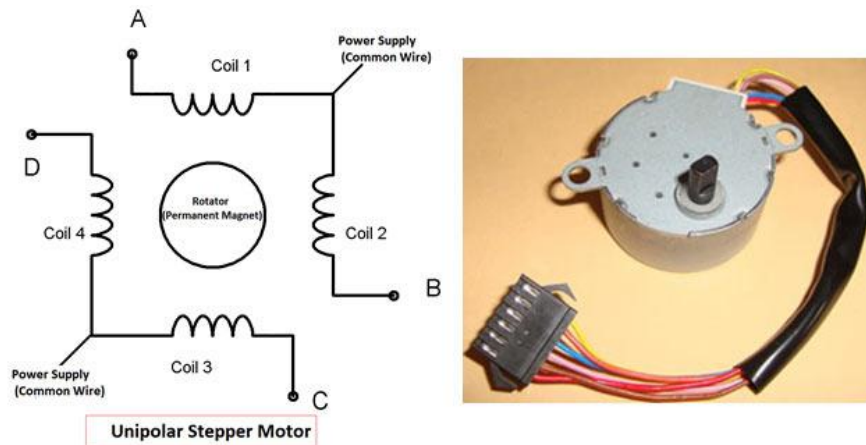
Raspberry Pi is an ARM architecture processor based board designed for electronic engineers and hobbyists. The PI is one of most trusted project development platforms out there now. With higher processor speed and 1 GB RAM, the PI can be used for many high profile projects like Image processing and Internet of Things.

For doing any of high profile projects, one need to understand the basic functions of PI. We will be covering all the basic functionalities of Raspberry Pi in these tutorials. In each tutorial we will discuss one of functions of PI. By the end of this Raspberry Pi Tutorial Series, you will be able to do high profile projects by yourself. Go through below tutorials:

- ❖ Getting Started with Raspberry Pi
- ❖ Raspberry Pi Configuration
- ❖ LED Blinky
- ❖ Raspberry Pi Button Interfacing
- ❖ Raspberry Pi PWM generation
- ❖ Controlling DC Motor using Raspberry Pi

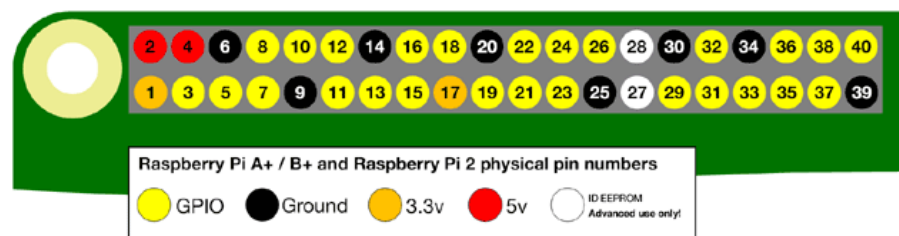
Here, we will Control the Speed of a Stepper Motor using Raspberry Pi. In Stepper Motor, as the name itself says, the rotation of shaft is in Step form. There are different types of

Stepper Motor; in here we will be using the most popular one that is Unipolar Stepper Motor. Unlike DC motor, we can rotate stepper motor to any particular angle by giving it proper instructions.



To rotate this Four Stage Stepper Motor, we will deliver power pulses by using Stepper Motor Driver Circuit. The driver circuit takes logic triggers from PI. If we control the logic triggers, we control the power pulses and hence the speed of stepper motor.

There are 40 GPIO output pins in Raspberry Pi 2. But out of 40, only 26 GPIO pins (GPIO2 to GPIO27) can be programmed. Some of these pins perform some special functions. With special GPIO put aside, we have only 17 GPIO remaining. Each of these 17 GPIO pin can deliver a maximum of 15mA current. And the sum of currents from all GPIO Pins cannot exceed 50mA. To know more about GPIO pins, go through: LED Blinking with Raspberry Pi



There are +5V (Pin 2 & 4) and +3.3V (Pin 1 & 17) power output pins on the board for connecting other modules and sensors. These power rails cannot be used to drive the Stepper Motor, because we need more power to rotate it. So we have to deliver the power to Stepper Motor from another power source. My stepper motor has a voltage rating of 9V so I am using a

9v battery as my second power source. Search your stepper motor model number to know voltage rating. Depending on the rating choose the secondary source appropriately.

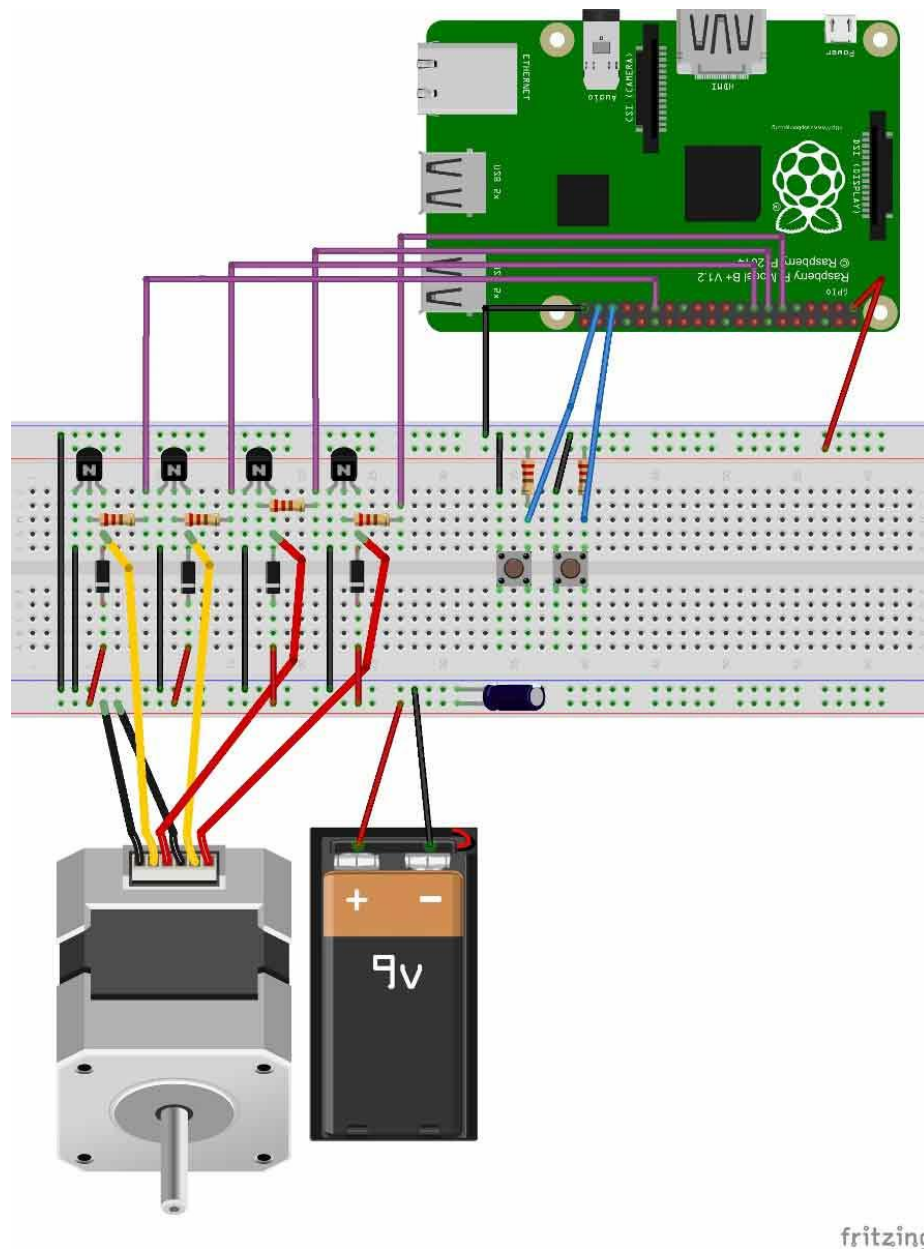
As stated earlier, we need a driver circuit to drive the Stepper Motor. We will also be designing a Simple Transistor Driver Circuit here.

### **Components Required:**

Here we are using Raspberry Pi 2 Model B with Raspbian Jessie OS. All the basic Hardware and Software requirements are previously discussed, you can look it up in the Raspberry Pi Introduction, other than that we need:

- ✓ Connecting pins
- ✓ 220 $\Omega$  or 1K $\Omega$  resistor (3)
- ✓ Stepper Motor
- ✓ Buttons (2)
- ✓ 2N2222 Transistor (4)
- ✓ 1N4007 Diode (4)
- ✓ Capacitor- 1000uF
- ✓ Bread Board

## Circuit Explanation:



Stepper motor use 200 steps to complete 360 degree rotation, means its rotate 1.8 degree per step. As we are driving a Four Stage Stepper Motor, so we need to give four pulses to complete single logic cycle. Each step of this motor completes 1.8 degree of rotation, so in order to complete a cycle we need 200 pulses. So  $200/4 = 50$  logic cycles needed to complete a single rotation. Check this to know more about Steppers Motors and its Driving Modes.

We will be driving each of these four coils by a NPN transistor (2N2222), this NPN transistor takes the logic pulse from PI and drives the corresponding coil. Four transistors are taking four logics from PI to drive four stages of stepper motor.

The transistor driver circuit is a tricky setup; here we should pay attention that wrongly connecting the transistor might load the board heavily and damage it. Check this to properly understand the Stepper Motor Driver Circuit.

The motor is an induction and so while switching the motor, we experience inductive spiking. This spiking will heat up the transistor heavily, so we will be using Diode (1N4007) to provide protection to transistor against Inductive Spiking.

In order to reduce the voltage fluctuations, we will be connecting a 1000uF capacitor across the power supply as shown in the Circuit Diagram.

### **Working Explanation:**

Once everything is connected as per the circuit diagram, we can turn ON the PI to write the program in PYHTON.

We will talk about few commands which we are going to use in PYHTON program,

We are going to import GPIO file from library, below function enables us to program GPIO pins of PI. We are also renaming “GPIO” to “IO”, so in the program whenever we want to refer to GPIO pins we will use the word ‘IO’.

```
import RPi.GPIO as IO
```

Sometimes, when the GPIO pins, which we are trying to use, might be doing some other functions. In that case, we will receive warnings while executing the program. Below command tells the PI to ignore the warnings and proceed with the program.

```
IO.setwarnings(False)
```



We can refer the GPIO pins of PI, either by pin number on board or by their function number. Like 'PIN 35' on the board is 'GPIO19'. So we tell here either we are going to represent the pin here by '35' or '19'.

```
IO.setmode (IO.BCM)
```

We are setting four of GPIO pins as output for driving four coils of stepper motor.

```
IO.setup(5,IO.OUT)
```

```
IO.setup(17,IO.OUT)
```

```
IO.setup(27,IO.OUT)
```

```
IO.setup(22,IO.OUT)
```

We are setting GPIO26 and GPIO19 as input pins. We will detect button press by these pins.

```
IO.setup(19,IO.IN)
```

```
IO.setup(26,IO.IN)
```

In case the Condition in the braces is true, the statements inside the loop will be executed once. So if the GPIO pin 26 goes low, then the statements inside the IF loop will be executed once. If the GPIO pin 26 does not goes low, then the statements inside the IF loop will not be executed.

```
if(IO.input(26) == False):
```

This command executes the loop 100 times, x being incremented from 0 to 99.

```
for x in range (100):
```

While 1: is used for infinity loop. With this command the statements inside this loop will be executed continuously.

We have all the commands needed to achieve the Speed Control of Stepper Motor with this.

After writing the program and executing it, all there is left is operating the control. We have two buttons connected to PI. One for increments the delay between the four pulses and other for decrements the delay between the four pulses. The delay itself speaks of speed; if the delay is higher the motor takes brakes between each step and so rotation is slow. If the delay is near zero, then the motor rotates at maximum speed.

Here it should be remember that, there should be some delay between the pulses. After giving a pulse, stepper motor takes few milliseconds of time to reach its final stage. If there is no delay given between the pulses, the stepper motor will not move at all. Normally 50ms delay is fine between the pulses. For more accurate information, look into the data sheet.

So with two buttons we can control the delay, which in turns control the speed of the stepper motor.

## Code

```
import RPi.GPIO as IO # we are calling for header file which helps us use GPIO's of PI
import time # we are calling for time to provide delays in program
IO.setwarnings(False) # do not show any warnings
x=1 # integer for storing the delay multiple
IO.setmode (IO.BCM)
IO.setup(5,IO.OUT) # initialize GPIO5 as an output.
IO.setup(17,IO.OUT)
IO.setup(27,IO.OUT)
IO.setup(22,IO.OUT)
IO.setup(19,IO.IN) # initialize GPIO19 as an input.
IO.setup(26,IO.IN)
while 1: # execute loop forever
 IO.output(5,1) # Step1 go high
 IO.output(22,0)
```

```

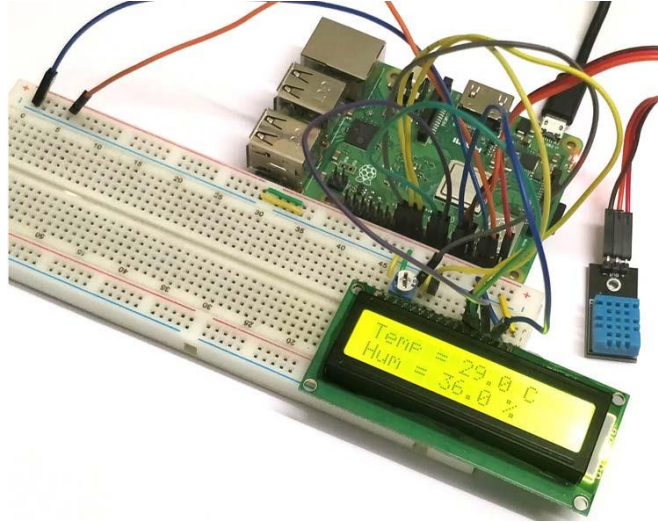
for y in range(x): # sleep for x*100msec
 time.sleep(0.01)
IO.output(17,1) # step2 go high
IO.output(5,0)
for y in range(x):
 time.sleep(0.01) # sleep for x*100msec
IO.output(27,1) #step 3 go high
IO.output(17,0)
for y in range(x):
 time.sleep(0.01) # sleep for x*100msec
IO.output(22,1) #step 4 go high
IO.output(27,0)
for y in range(x):
 time.sleep(0.01) # sleep for x*100msec
 if(IO.input(26) == False): #if button1 is pressed
 if(x<100):
 x=x+1 #increment x by one if x<100
 time.sleep(0.5) #sleep for 500ms
 if(IO.input(19) == False): #if button2 is pressed
 if(x>1):
 x=x-1 #decrement x by one if x>1
 time.sleep(0.5) #sleep for 500ms

```

## Project – 06

# Interfacing DHT11 Temperature and Humidity Sensor with Raspberry Pi

---



Temperature and Humidity are the most common parameters that are being monitored in any environment. There are tons of sensors to choose from for measuring temperature and humidity, but the most used one is the DHT11 due to its decent measuring range and accuracy. It also works with one pin communication and hence is very easy to interface with Microcontrollers or Microprocessors. In this tutorial we are going to learn how to interface the popular DHT11 sensor with Raspberry Pi and display the value of temperature and humidity on a 16x2 LCD screen. We already used it to build IoT Raspberry Pi Weather Station.

### **Overview of DHT11 Sensor:**

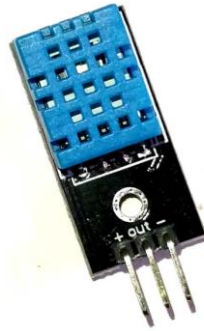
The DHT11 sensor can measure relative humidity and temperature with the following specifications

Temperature Range: 0-50°C

Temperature Accuracy:  $\pm 2$  °C

Humidity Range: 20-90% RH

Humidity Accuracy:  $\pm 5$  %



The DHT11 sensor is available either in module form or in sensor form. In this tutorial we are using the module form of the sensor, the only difference between both is that in module form the sensor has a filtering capacitor and a pull up resistor attached to the output pin of the sensor. So if you are using the sensor alone make sure you add these two components. Also learn DHT11 interfacing with Arduino.

### **How DHT11 Sensor works:**

The DHT11 sensor comes with a blue or white colour casing. Inside this casing we have two important components which help us to sense the relative humidity and temperature. The first component is a pair of electrodes; the electrical resistance between these two electrodes is decided by a moisture holding substrate. So the measured resistance is inversely proportional to the relative humidity of the environment. Higher the relative humidity lower will be the value of resistance and vice versa. Also note that Relative humidity is different from actual humidity. Relative humidity measures the water content in air relative to the temperature in the air.

The other component is a surface mounted NTC Thermistor. The term NTC stands for Negative temperature coefficient, for increase in temperature the value of resistance will decrease

### **Pre-Requisites:**

It is assumed that your Raspberry Pi is already flashed with an operating system and is able to connect to the internet. If not, follow the Getting started with Raspberry Pi tutorial before proceeding.

It is also assumed that you have access to your pi either through terminal windows or through other application through which you can write and execute python programs and use the terminal window.

### **Installing the Adafruit LCD library on Raspberry Pi:**

The value of the temperature and humidity will be displayed on a 16\*2 LCD display. Adafruit provides us a library to easily operate this LCD in 4-bit mode, so let us add it to our Raspberry Pi by opening the terminal window Pi and following the below steps.

**Step 1:** Install git on your Raspberry Pi by using the below line. Git allows you to clone any project files on Github and use it on your Raspberry pi. Our library is on Github so we have to install git to download that library into pi.

```
apt-get install git
```

**Step 2:** The following line links to the GitHub page where the library is present just execute the line to clone the project file on Pi home directory

```
git clone git://github.com/adafruit/Adafruit_Python_CharLCD
```

**Step 3:** Use the below command to change directory line, to get into the project file that we just downloaded. The command line is given below

```
cd Adafruit_Python_CharLCD
```

**Step 4:** Inside the directory there will be a file called setup.py, we have to install it, to install the library. Use the following code to install the library

```
sudo python setup.py install
```

That is it the library should have been installed successfully. Now similarly let's proceed with installing the DHT library which is also from Adafruit.

## Installing the Adafruit DHT11 library on Raspberry Pi:

DHT11 Sensor works with the principle of one-wire system. The value of temperature and humidity is sensed by the sensor and then transmitted through the output pin as serial data. We can then read these data by using I/O pin on a MCU/MPU. To understand how these values are read you would have to read through the datasheet of the DHT11 sensor, but for now to keep things simple we will use a library to talk with the DHT11 sensor.

The DHT11 library provided by Adafruit can be used for DHT11, DHT22 and other one wire temperature sensors as well. The procedure to install the DHT11 library is also similar to the one followed for installing LCD library. The only line that would change is the link of the GitHub page on which the DHT library is saved.

Enter the four command lines one by one on the terminal to install the DHT library

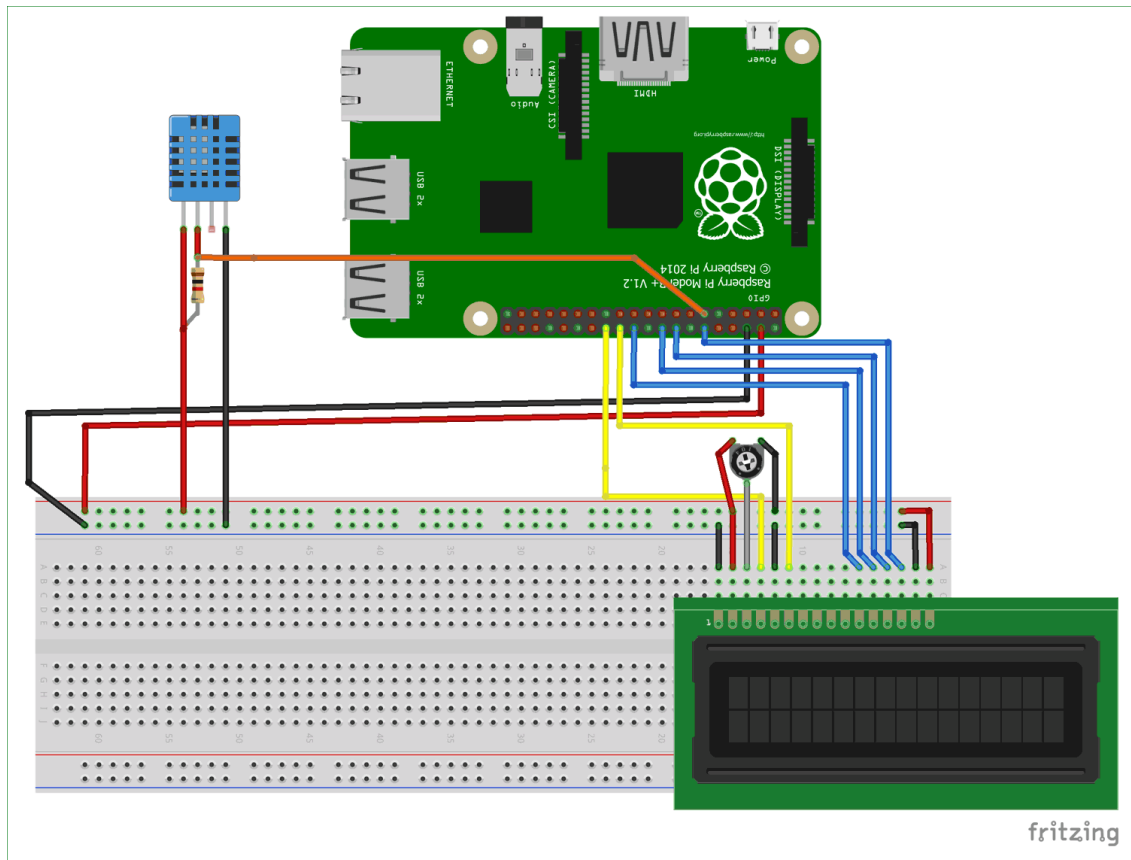
```
git clone https://github.com/adafruit/Adafruit_Python_DHT.git
```

```
cd Adafruit_Python_DHT
sudo apt-get install build-essential python-dev
sudo python setup.py install
```

Once it is done you will have both the libraries successfully installed on our Raspberry Pi. Now we can proceed with the hardware connection.

## Circuit Diagram:

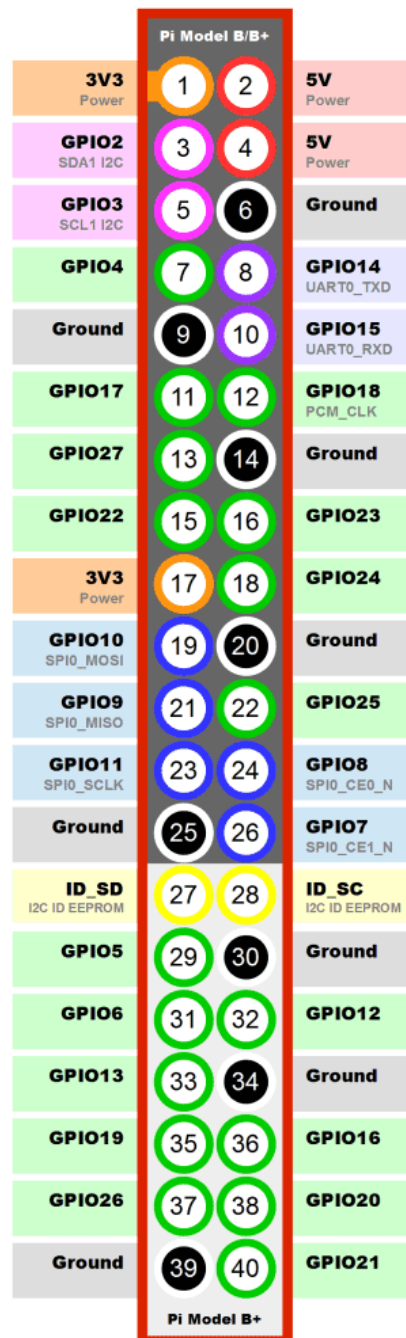
The complete circuit diagram Interfacing DH11 with Raspberry pi is given below, it was built using Fritzing. Follow the connections and make the circuit



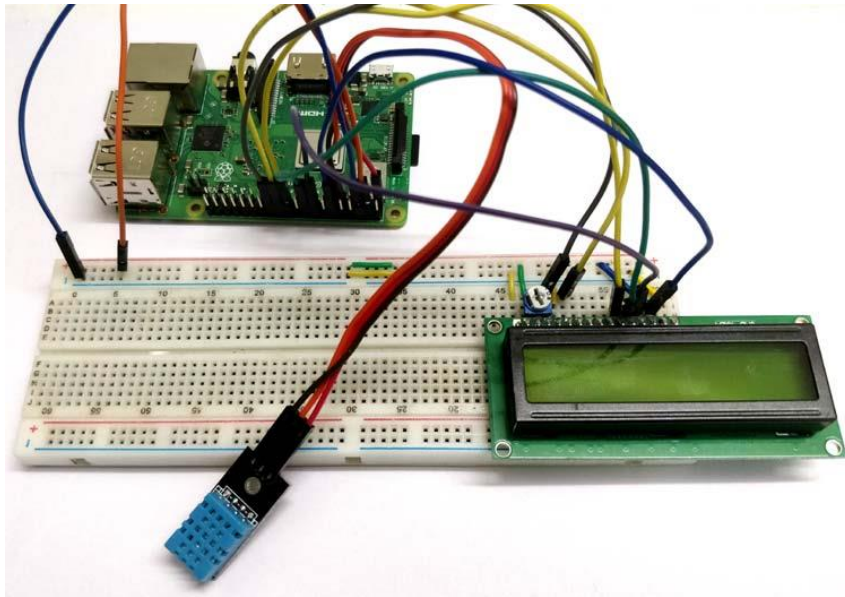
Both the LCD and DHT11 sensor works with +5V supply so we use the 5V pins on the Raspberry Pi to power both. A pull up resistor of value 1k is used on the output pin of the DHT11 sensor, if you are using a module you can avoid this resistor.

A trimmer pot of 10k is added to the Vee pin of the LCD to control the contrast level of the LCD. Other than that all the connections are pretty straight forward. But make a note of which GPIO pins you are using to connect the pins since we will need in our program. The below chart should allow you to figure out the GPIO pin numbers.





Use the chart and make your connections according to the circuit diagram. I used a breadboard and jumper wires to make my connections. Since I used DHT11 module I wired it directly to Raspberry Pi. My hardware looked like this below



### Python Programming for DHT11 sensor:

We have to write a program to read the value of temperature and humidity from the DHT11 sensor and then display the same on the LCD. Since we have downloaded libraries for both LCD and DHT11 sensor the code should be pretty much straight forward. The python complete program can be found at the end of this page, but you can read further to understand how the program works.

We have to import the LCD library and DHT11 library into our program to use the functions related to it. Since we have already downloaded and installed them on our Pi we can simply use the following lines to import them. We also import the time library to use the delay function.

```
import time #import time for creating delay

import Adafruit_CharLCD as LCD #Import LCD library

import Adafruit_DHT #Import DHT Library for sensor
```

Next, we have to specify to which pins the sensor is connected to and what type of temperature sensor is used. The variable `sensor_name` is assigned to `Adafruit_DHT.DHT11` since we are using the DHT11 sensor here. The output pin of the sensor is connected to GPIO 17 of the Raspberry Pi and hence we assign 17 to `sensor_pin` variable as shown below.

```
sensor_name = Adafruit_DHT.DHT11 #we are using the DHT11 sensor
sensor_pin = 17 #The sensor is connected to GPIO17 on Pi
```

Similarly, we also have to define to which GPIO pins the LCD is connected to. Here we are using the LCD in 4-bit mode hence we will have four data pins and two control pins to connect to the GPIO pins of the pi. Also, you can connect the backlight pin to a GPIO pin if we wish to control the backlight also. But for now I am not using that so I have assigned 0 to it.

```
lcd_rs = 7 #RS of LCD is connected to GPIO 7 on PI
lcd_en = 8 #EN of LCD is connected to GPIO 8 on PI
lcd_d4 = 25 #D4 of LCD is connected to GPIO 25 on PI
lcd_d5 = 24 #D5 of LCD is connected to GPIO 24 on PI
lcd_d6 = 23 #D6 of LCD is connected to GPIO 23 on PI
lcd_d7 = 18 #D7 of LCD is connected to GPIO 18 on PI
lcd_backlight = 0 #LED is not connected so we assign to 0
```

You can also connect LCD in 8-bit mode with Raspberry pi but then free pins will be reduced.

The LCD library from Adafruit that we downloaded can be used for all types of characteristic LCD displays. Here in our project we are using a 16\*2 LCD display so we are mentioning the number of Rows and Columns to a variable as shown below.

```
lcd_columns = 16 #for 16*2 LCD
lcd_rows = 2 #for 16*2 LCD
```

Now, that we have declared the LCD pins and the number of Rows and Columns for the LCD we can initialize the LCD display by using the following line which sends all the required information to the library.

```
lcd = LCD.Adafruit_CharLCD(lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7,
```

```
lcd_columns, lcd_rows, lcd_backlight) #Send all the pin details to library
```

To start the program, we display a small intro message using the `lcd.message()` function and then give a delay of 2 second to make the message readable. For printing on the 2<sup>nd</sup> line the command `\n` can be used as shown below

```
lcd.message('DHT11 with Pi \n -CircuitDigest') #Give a intro message
time.sleep(2) #wait for 2 secs
```

Finally, inside our while loop we should read the value of temperature and humidity from the sensor and display it on the LCD screen for every 2 seconds. The complete program inside the while loop is shown below

```
while 1: #Infinite Loop
```

```
humidity, temperature = Adafruit_DHT.read_retry(sensor_name, sensor_pin) #read from sensor
and save respective values in temperature and humidity varibale
```

```
lcd.clear() #Clear the LCD screen

lcd.message ('Temp = %.1f C' % temperature) # Display the value of temperature

lcd.message ('\nHum = %.1f %%' % humidity) #Display the value of Humidity

time.sleep(2) #Wait for 2 sec then update the values
```

We can easily get the value of temperature and humidity from the sensor using this single line below. As you can see it return two values which is stored in the variable humidity and temperature. The `sensor_name` and `sensor_pin` details are passed as parameters; these values were updated in the beginning of the program

```
humidity, temperature = Adafruit_DHT.read_retry(sensor_name, sensor_pin)
```

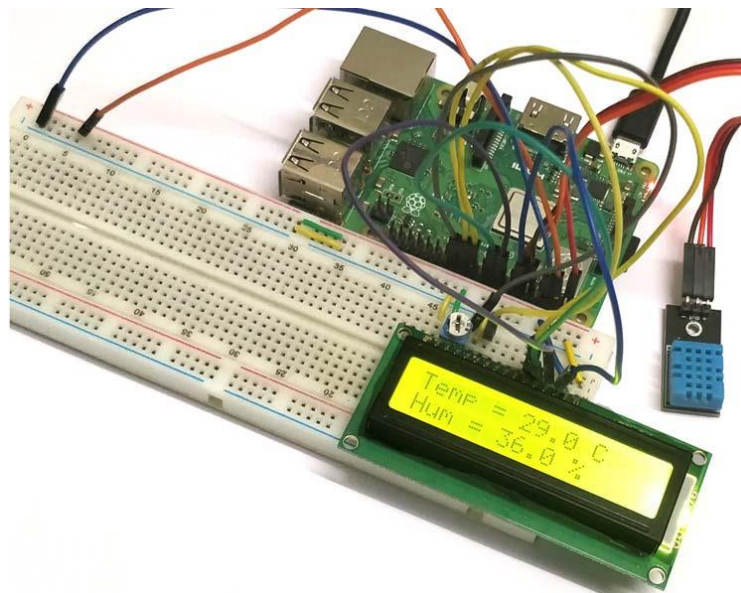
To display a variable name on the LCD screen we can use the identifiers like &d, %c etc. Here since we are displaying a floating point number with only one digit after the decimal point we use the identifier %.1f for displaying the value in the variable temperature and humidity

```
lcd.message ('Temp = %.1f C' % temperature)

lcd.message ('\nHum = %.1f %%' % humidity)
```

### Measuring Humidity and Temperature using Raspberry Pi:

Make the connections as per the circuit diagram and install the required libraries. Then launch the python program given at the end of this page. Your LCD should display an intro message and then display the current temperature and humidity value as shown in the image below.



If you find nothing being displayed the LCD, check if the python shell window is displaying any errors, if no error is displayed then check your connections once more and adjust the potentiometer to vary the contrast level of the LCD and check if you get anything on the screen.

Hope you understood the project and enjoyed building it, if you has faced any problem in getting this done report it on the comment section or use the forum for technical help. I will try my best to respond to all comments.

## Code

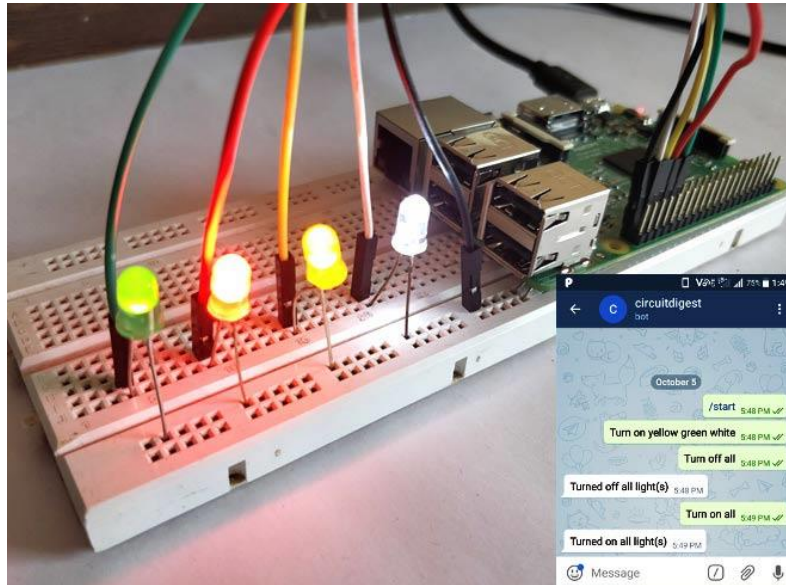
#Program to read the values of Temp and Hum from the DHT11 sensor and display them on the LCD

#Website: [www.circuitdigest.com](http://www.circuitdigest.com)

```
import time #import time for creating delay
import Adafruit_CharLCD as LCD #Import LCD library
import Adafruit_DHT #Import DHT Library for sensor
sensor_name = Adafruit_DHT.DHT11 #we are using the DHT11 sensor
sensor_pin = 17 #The sensor is connected to GPIO17 on Pi
lcd_rs = 7 #RS of LCD is connected to GPIO 7 on PI
lcd_en = 8 #EN of LCD is connected to GPIO 8 on PI
lcd_d4 = 25 #D4 of LCD is connected to GPIO 25 on PI
lcd_d5 = 24 #D5 of LCD is connected to GPIO 24 on PI
lcd_d6 = 23 #D6 of LCD is connected to GPIO 23 on PI
lcd_d7 = 18 #D7 of LCD is connected to GPIO 18 on PI
lcd_backlight = 0 #LED is not connected so we assign to 0
lcd_columns = 16 #for 16*2 LCD
lcd_rows = 2 #for 16*2 LCD
lcd = LCD.Adafruit_CharLCD(lcd_rs, lcd_en, lcd_d4, lcd_d5, lcd_d6, lcd_d7,
 lcd_columns, lcd_rows, lcd_backlight) #Send all the pin details to library
lcd.message('DHT11 with Pi \n -CircuitDigest') #Give a intro message
time.sleep(2) #wait for 2 secs
while 1: #Infinite Loop
 humidity, temperature = Adafruit_DHT.read_retry(sensor_name, sensor_pin) #read from sensor and save respective values in temperature and humidity variables
 lcd.clear() #Clear the LCD screen
 lcd.message('Temp = %.1f C' % temperature) # Display the value of temperature
 lcd.message('\nHum = %.1f %' % humidity) #Display the value of Humidity
 time.sleep(2) #Wait for 2 sec then update the values
```

## Project – 07

## Controlling Raspberry Pi GPIO Pins using Telegram App



Telegram is an optimal application to combine with Raspberry Pi for all our mobile control purpose. It has very good developer support and lots of features are being planned to be released soon to enhance the performance of Telegram Bots. In our previous tutorial we learnt how we can set up a telegram bot for raspberry pi and also learnt have to have a chat with it and share images, documents and Audio files.

Now, we will proceed to next step by learning How we can control the GPIO pins on Raspberry Pins using Telegram, so that we provide some hardware support for our bot. In this tutorial we will Connect four LEDs to Raspberry Pi GPIO pins and toggle them using natural language (chatting like) from Telegram. Sounds interesting right? Let us get started.

### **Materials Required:**

1. Four LED (any color)
2. Raspberry Pi (with internet connection)
3. Breadboard
4. Connecting wires

### **Pre-Requisites:**

Before proceeding with the project make sure your Raspberry Pi is connected to internet and you can run python programs on your Pi. Also know how to set up Telegram bot with Raspberry Pi, since I will assume you are familiar with that stuff to proceed with the project.

### **Circuit Diagram:**

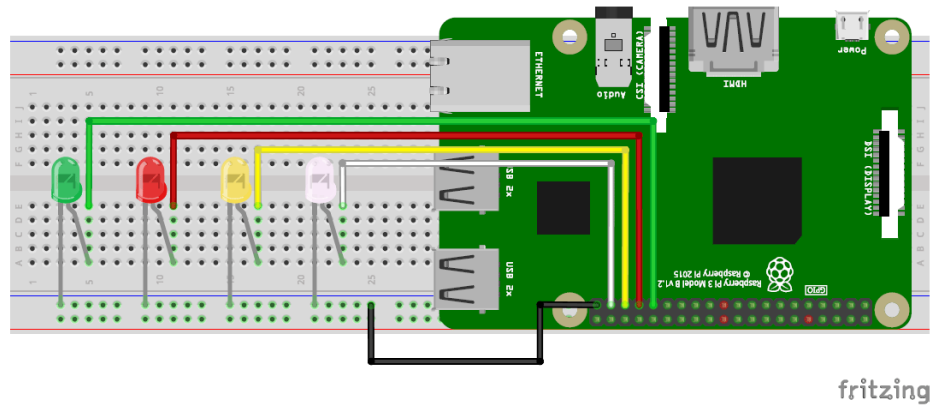
The circuit Diagram for Controlling LEDs using Raspberry Pi and Telegram Android App is nothing more than four LEDs and some connecting wires. We will not need the current limiting resistors since the Raspberry Pi GPIO pins work on 3.3V TTL. Follow the circuit below and connect your LED.

The following table will help you determine the pin number and GPIO number for the connection of four leds.

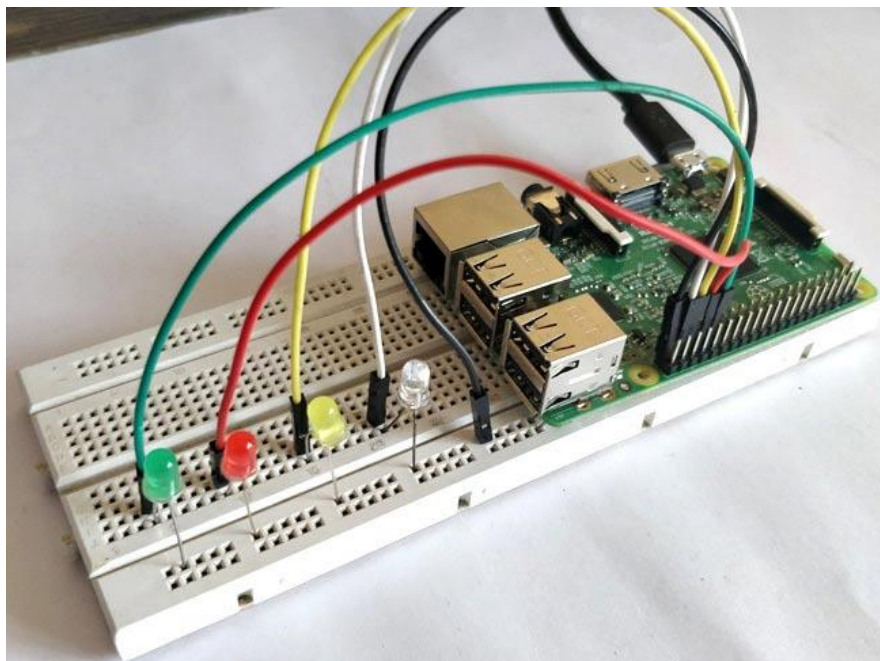
| Led Terminal        | Pin Number | GPIO Number |
|---------------------|------------|-------------|
| Green Anode         | Pin 31     | GPIO 6      |
| Red Anode           | Pin 33     | GPIO 13     |
| Yellow Anode        | Pin 35     | GPIO 19     |
| White Anode         | Pin 37     | GPIO 26     |
| Cathode of all four | Pin 39     | Ground      |

Below is the Circuit diagram in which four LEDs are connected according to the Table given above:





Once your connections your hardware set-up should look like something like this below.



### **Raspberry Python Program:**

Once the hardware is ready, we can proceed with the Python Program. In this program we have to read the data (message) sent from the Telegram bot and toggle the LED accordingly. To make it more natural, instead of checking each sentence and hard coding those sentence inside our program we can check for words and proceed accordingly.

So the program will primarily check for two words, they are on and off. Once detecting either one of these two words, it will look for other keywords like white, yellow, green and red. The respective colour LED will be toggled only if the word is detected. We will also update a string for the detected words to send a message back to telegram bot.

The complete program can be found at the bottom of this page; just below I have explained the program by breaking it into small meaningful junks.

For this program to work, we need the telepot downloaded and imported in our Raspberry Pi. In our previous tutorial we have already downloaded the telepot inside our Raspberry Pi, so now we just have to import it into our program along with the GPIO library as shown below.

```
import RPi.GPIO as GPIO

import telepot

from telepot.loop import MessageLoop
```

We will be controlling for LED lights using this program and the colour of the LEDs will be White, Yellow, Red and Green. They are connected to the pins shown in circuit diagram; let us define the pin names for these LEDs based on their colour so that it is use them in the program.

```
white = 26

yellow = 19

red = 13

green = 6
```

The next step would be to define all these LED pins as output pins and define them as turned off by default by using the below lines.

```
#LED White

GPIO.setup(white, GPIO.OUT)

GPIO.output(white, 0) #Off initially

#LED Yellow

GPIO.setup(yellow, GPIO.OUT)

GPIO.output(yellow, 0) #Off initially

#LED Red
```

```
GPIO.setup(red, GPIO.OUT)

GPIO.output(red, 0) #Off initially

#LED green

GPIO.setup(green, GPIO.OUT)

GPIO.output(green, 0) #Off initially
```

As we learnt in our previous tutorial all the actions that has to be done by the Raspberry bot will be defined inside the function action. Here we have to make the bot to listen to the message send from mobile, compare it to some keywords and toggle LED accordingly.

For each message we send from mobile, there will be a chat id and command. This chat id is required by the program to reply back to the sender. So we save the chat id and, message as shown below.

```
chat_id = msg['chat']['id']

command = msg['text']
```

Now, whatever we send from the phone will be saved as string in the variable command. So, all we have to do is check for key words in this variable. Python has a command make things easy here. For example, if we have to check if the word “on” is present in the string stored in command variable we can simply use the below line.

```
if 'on' in command:
```

Similarly we check for all keywords, once we receive an “on”, we proceed to check for which colour the user has mentioned. This is also done by the same commands by comparing the same keywords. We also update a string named message that can be replied back to the user as a status message.

```
if 'on' in command:

 message = "Turned on "
```

```

if 'white' in command:

 message = message + "white "

 GPIO.output(white, 1)

if 'yellow' in command:

 message = message + "yellow "

 GPIO.output(yellow, 1)

if 'red' in command:

 message = message + "red "

 GPIO.output(red, 1)

if 'green' in command:

 message = message + "green "

 GPIO.output(green, 1)

if 'all' in command:

 message = message + "all "

 GPIO.output(white, 1)

 GPIO.output(yellow, 1)

 GPIO.output(red, 1)

 GPIO.output(green, 1)

message = message + "light(s)"

telegram_bot.sendMessage (chat_id, message)

```

As shown above we look for keywords like ‘green’, ‘white’, ‘red’, ‘yellow’ and ‘all’ and ‘Turned on’ that particular LED alone. Once the job is done we send a message back to the user about what just happened. The same method can be used to turn off the lights off well.

```

if 'off' in command:

 message = "Turned off "

```

```

if 'white' in command:

 message = message + "white "

 GPIO.output(white, 0)

if 'yellow' in command:

 message = message + "yellow "

 GPIO.output(yellow, 0)

if 'red' in command:

 message = message + "red "

 GPIO.output(red, 0)

if 'green' in command:

 message = message + "green "

 GPIO.output(green, 0)

if 'all' in command:

 message = message + "all "

 GPIO.output(white, 0)

 GPIO.output(yellow, 0)

 GPIO.output(red, 0)

 GPIO.output(green, 0)

message = message + "light(s)"

telegram_bot.sendMessage (chat_id, message)

```

### **Controlling LEDs with Raspberry Pi and Telegram bot:**

Connect your LEDs and launch your program on python. Make sure you have changed the Token address for your bot. And start typing in the commands you wish. For example to turn on the red and yellow light you can use any of the following command.

#### **1.Turn on Red and Yellow Light**

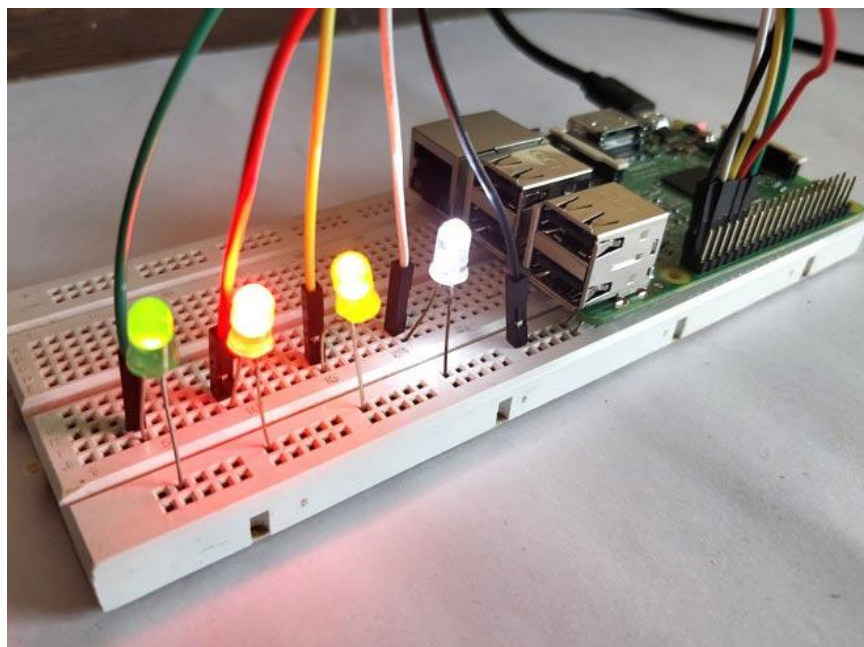
2.Switch on Red and Yellow colour right

3.On red and yellow

4.Please put on the yellow and red light

What not.....

As you can see the bot only looks for the Keywords and will ignore the other words in the Sentence, this way you can speak to it naturally. The complete working of the project can be found at the Video given at the end of this page.



Go ahead! play with your project and have fun. You can take it to a whole new level now. With both the tutorial combined we have the power to control any hardware from our Smart phone anywhere from the world and also get inputs/results from our Raspberry Pi in form of message, Audio, Image and even as document. If you replace the LEDs with Relays and AC appliances, then it could be a Smart Phone controlled Home Automation. So, use your creativity and build your own cool projects...

## Code

```
import time, datetime
import RPi.GPIO as GPIO
import telepot
from telepot.loop import MessageLoop
white = 26
yellow = 19
red = 13
green = 6
now = datetime.datetime.now()
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
#LED White
GPIO.setup(white, GPIO.OUT)
GPIO.output(white, 0) #Off initially
#LED Yellow
GPIO.setup(yellow, GPIO.OUT)
GPIO.output(yellow, 0) #Off initially
#LED Red
GPIO.setup(red, GPIO.OUT)
GPIO.output(red, 0) #Off initially
#LED green
GPIO.setup(green, GPIO.OUT)
GPIO.output(green, 0) #Off initially
```

```

def action(msg):
 chat_id = msg['chat']['id']
 command = msg['text']
 print 'Received: %s' % command
 if 'on' in command:
 message = "Turned on "
 if 'white' in command:
 message = message + "white "
 GPIO.output(white, 1)
 if 'yellow' in command:
 message = message + "yellow "
 GPIO.output(yellow, 1)
 if 'red' in command:
 message = message + "red "
 GPIO.output(red, 1)
 if 'green' in command:
 message = message + "green "
 GPIO.output(green, 1)
 if 'all' in command:
 message = message + "all "
 GPIO.output(white, 1)
 GPIO.output(yellow, 1)
 GPIO.output(red, 1)
 GPIO.output(green, 1)
 message = message + "light(s)"
 telegram_bot.sendMessage(chat_id, message)
 if 'off' in command:
 message = "Turned off "
 if 'white' in command:
 message = message + "white "
 GPIO.output(white, 0)
 if 'yellow' in command:

```



```

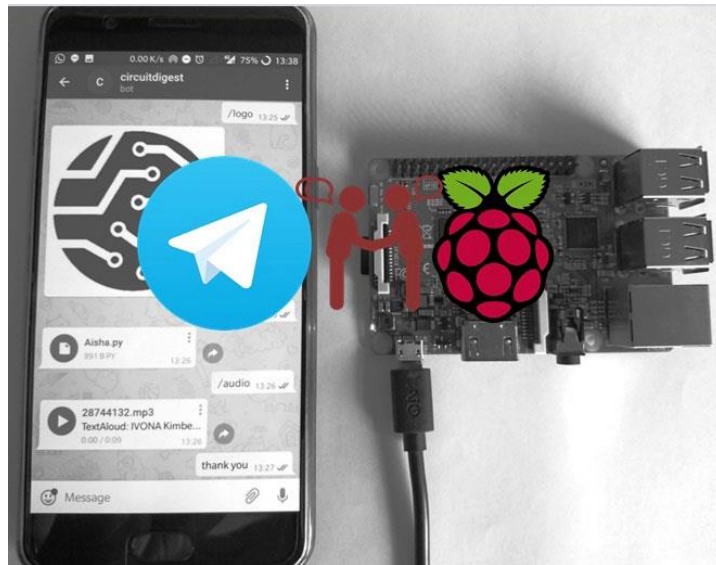
 message = message + "yellow "
 GPIO.output(yellow, 0)
 if 'red' in command:
 message = message + "red "
 GPIO.output(red, 0)
 if 'green' in command:
 message = message + "green "
 GPIO.output(green, 0)
 if 'all' in command:
 message = message + "all "
 GPIO.output(white, 0)
 GPIO.output(yellow, 0)
 GPIO.output(red, 0)
 GPIO.output(green, 0)
 message = message + "light(s)"
 telegram_bot.sendMessage(chat_id, message)
telegram_bot = telepot.Bot('470583174:AAG7MPZc93qchp-tjqA_K2meRYcQiOR7X7Y')
print (telegram_bot.getMe())
MessageLoop(telegram_bot, action).run_as_thread()
print 'Up and Running....'
while 1:
 time.sleep(10)

```

### **Project – 08**

## **Sharing Text and Files: Using Telegram Bot with Raspberry Pi**

---



Raspberry Pi has always been fun and easy to build projects. Its powerful ARM architecture and open-source Linux based Operating System has help us a lot in getting our projects online in no time. In this tutorial we will learn another interesting way to share data (files/photos/videos/audios/text) between Raspberry Pi and our Mobile phone through a popular chat application called Telegram.

For those who are new to Telegram, it is a chat based application available in play store for Android (also available for Iphone and windows) that is very similar to Whatsapp. It has over 100 million downloads (as on 5-10-2017) on play store and people claim it to be faster and more functional than Whatsapp (fingers crossed). One special features of this application is that they support bots. Meaning this smart phone application can not only be used by Humans but also by machine. In our case the machine will be Raspberry Pi. Once you train Raspberry Pi on how to act as a bot, anyone (if you make it public) can chat with your Raspberry Pi like chatting to any normal person and even share Photos Pictures Documents and Audio files. You can even train it to be your own Personal assistant, sounds cool right? Lets learn how to build a Raspberry pi telegram bot.

### **Materials Required:**

1. Any Raspberry Pi connected to Internet
2. A mobile running Telegram Application.

There is not much hardware involved in this project so relax on your chair with your Pi and follow the steps below. If you are new to Raspberry Pi then follow our Raspberry Pi Introduction article and other Raspberry Pi Tutorials.

**Note:** Make sure that your Pi is already connected to internet and you know how to use the Lx terminal on you Pi. So connect your Pi to internet before proceeding.

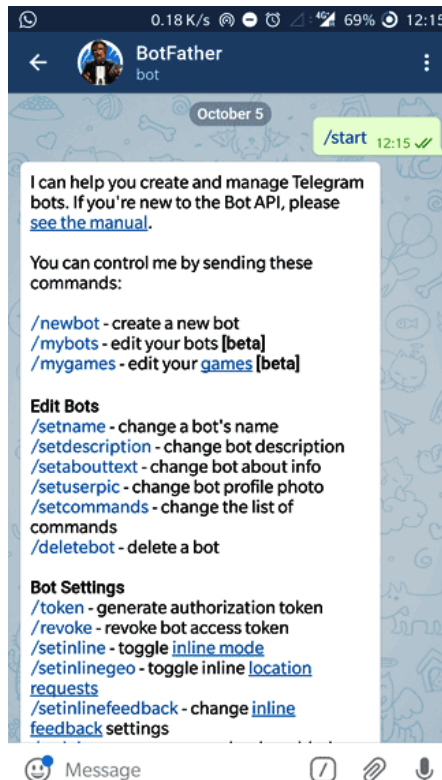


### ***Step 1: Installing Telegram on your Mobile***

The first step would be to install Telegram on your Mobile. Telegram is available for Android, IOS and even for Windows platform so just go ahead and download your Telegram application. Just like all application there will be a small Sign Up procedure to start using Telegram, continue with it until you reach your home screen.

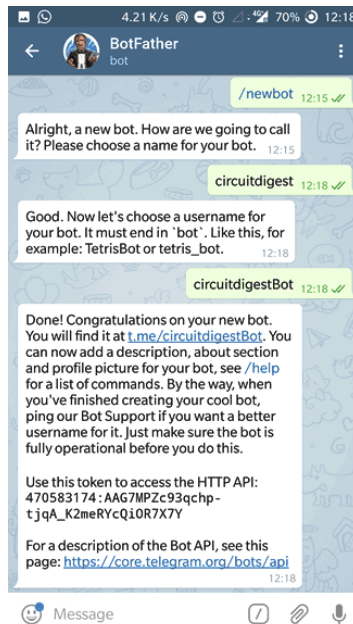
### ***Step2: Talk to Bot Father***

The next step would be to request the Bot Father to create us a new Bot. On the top right corner of the Home screen there will be a search icon, click on it to search for the name “botfather”. Botfather is a Bot by itself, it will guide you to create a new bot for you. Click on start and select /newbot as shown in the picture below. Now, the bot will ask for few details like name of your Bot and the user name of the bot. Fill those details and remember the username for we will needing it in future.



### ***Step3: Getting your token for access***

I have named bot as circuitdigest and the username as circuitdigestBot. After this process the botfather will give you a Token for access. This is like the password for your bot, people can control program your bot using this token key. So keep it safe and do not share it with anyone. Once you have received this token key it is time to move on to Raspberry Pi.



#### ***Step4: Telepot for installing Telegram on Raspberry Pi***

Using Telegram Bot in Raspberry Pi is made possible by the python package called Telepot. We need to install this package on Raspberry Pi by using the following commands on Lx terminal

```
sudo apt-get install python-pip
sudo pip install telepot
```

Once Telepot is imported into Raspberry we can use this package in our python program to communicate with our Telegram Bot.

#### ***Step 5: Programming your Raspberry Pi***

The bot that we just created on Telegram is just like a baby, it cannot do anything on its own unless we teach it what and how to do things. This teaching can be done through Raspberry Pi and Python script. In this tutorial I have programmed the bot to perform some basic actions like sending a message, Photo, Audio and Document. So when you say a particular command it will respond with a particular action the command and action is listed in the table below

| Command from Telegram | Response by Raspberry Pi                             |
|-----------------------|------------------------------------------------------|
| /hi                   | Replies with a string “Hi! CircuitDigest”            |
| /time                 | Replies with current time                            |
| /logo                 | Replies with an Image (logo of CircuitDigest)        |
| /file                 | Replies with a file ( that contains current program) |
| /audio                | Replies with a demo audio file                       |

The complete program to make the above actions is given at the bottom of this page. But just below, I have explained the important snippets in the program to help you understand how the program works.

The first step is to import all the libraries, here we will obviously need the telepot library to use the Telegram bot. We also make use of the time, datetime library to read the current time for Raspberry pi. Then we create a object now in which the value is stored.

```
import time, datetime

import telepot

from telepot.loop import MessageLoop

now = datetime.datetime.now()
```

The next step is to create a function for taking actions based on incoming commands from Telegram app on Mobile. Here the name of the function is action. It is inside this function where the bot comes to life. Our bot cannot initiate a conversation on its own, it can only reply if we ask something. So each time we ask something there will be chat id. This chat id is something similar to a address, only using this chat id a bot can reply back to us. So the first

step is to read the chat id and the message it is trying to say to us. We also print the received message for debugging purpose.

```
def action(msg):

 chat_id = msg['chat']['id']

 command = msg['text']

 print 'Received: %s' % command
```

Further down inside the function we compare this command with a predefined text and perform particular tasks. This first command will be /hi to which we reply “Hi! CircuitDigest”

```
if command == '/hi':

 telegram_bot.sendMessage (chat_id, str("Hi! CircuitDigest"))
```

The next command will be /time, to which we reply the current time. We already have the time and date in now, here simply split it based on hour and minute and add it as using the str function.

```
elif command == '/time':

 telegram_bot.sendMessage(chat_id, str(now.hour)+str(":")+str(now.minute))
```

The next command will be /logo, to which the bot will fetch an image from a url and send it to us. A image can be sent either from a URL or from the hard disk. Here I have just used the URL which displays the logo of CircuitDigest.

```
elif command == '/logo':

 telegram_bot.sendPhoto (chat_id, photo="https://i.pinimg.com/avatars/circuitdigest_1464122100_280.jpg")
```

The next command will be /file, which will send the file named Aisha.py from the hard disk. You can send any file that you wish to by changing the address of the directory

```
elif command == '/file':
```

```
 telegram_bot.sendDocument(chat_id, document=open('/home/pi/Aisha.py'))
```

The last command will be /audio. This can send any mp3 file from the hard disk, I have just used a audio file called test.mp3 as a demo here

```
elif command == '/audio':
```

```
 telegram_bot.sendAudio(chat_id, audio=open('/home/pi/test.mp3'))
```

Okay now comes the most important step, this is where we give access of our Telegram bot to the Python script. Here we name out bot as telegram\_bot and assign it the token address that was given by our botfather in step 3. In the line below I have removed the last few digits of my token as a matter of privacy. We also use the print get me to display the details of the Bot on the shell screen, this will help us notice things working.

```
telegram_bot = telepot.Bot('468382312:AAFhURMxpVIMWEdFzbIQLszBPFEUpXXXXXX')
print (telegram_bot.getMe())
```

Hope you understood how the program works, now let us move to next step.


### ***Step 6: Running the Program in your Raspberry Pi***

As said earlier the complete program is given at the end of this page, you can also download the same from here. Once you open the code make sure you change the token address of the program to your token address.

Now run the python code and you should see the details of your bot on the shell window like this

Here, my bot user name is circuitdigestBot. If you get your bots name here it means everything is going fine.



```
Python 2.7.13 Shell
File Edit Shell Debug Options Window Help
Python 2.7.13 (default, Jan 19 2017, 14:48:08)
[GCC 6.3.0 20170124] on linux2
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/pi/Aisha.py =====
{'username': u'circuitdigestBot', u'first_name': u'circuitdigest', u'is_bot': True, u'id': }
Up and Running...
```

Once you see “up and running” it means that your bot is ready for action and can now reply to your commands.

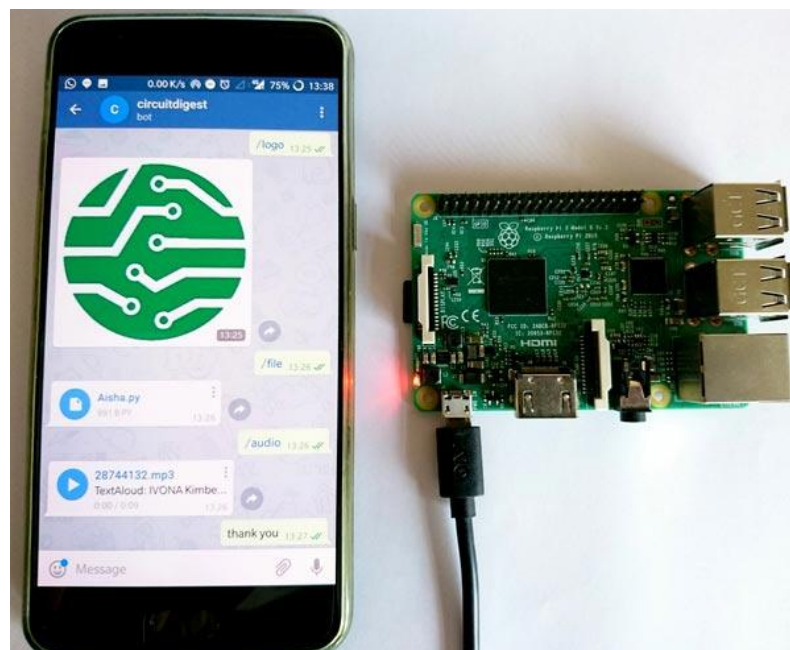
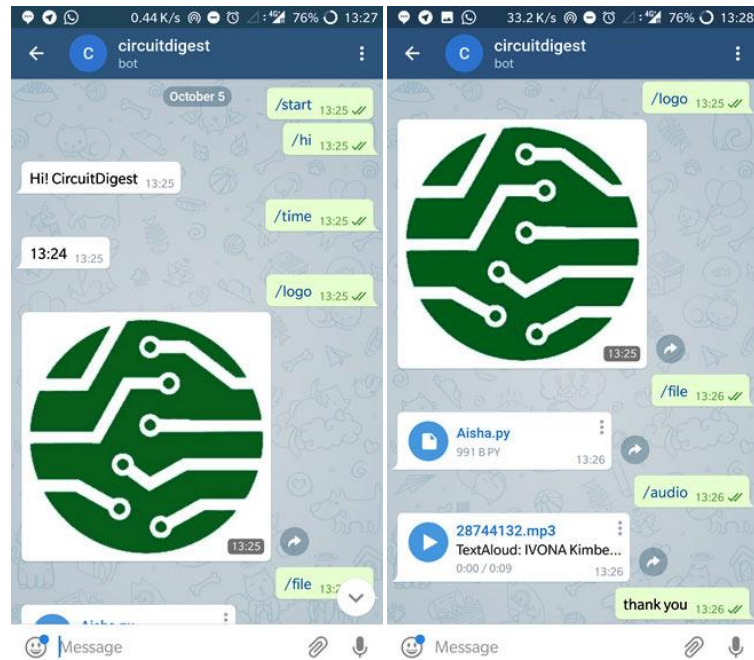
### ***Step 7: Enjoying the Output***

Now, all that is left is to check how good your bot is responding for your program. Search for your bot name in my case it is “circuitdigestBot”. Search for username and not or Bot name, your user name should end with bot.

Once you open your bot, click on start and type in any commands like /hi, /time, /file, /logo, or /audio and you should be replied accordingly.

**Note:** You might have problem with /audio and /file if you have not changed your directory to a proper file that is available on your Raspberry Pi.

You can use the shell script to see what your script is currently responding to. The complete working can be found at the video given at the end of this page.



### ***Step 8: Give me a High Five***

Hope you understood the project and now will be able create this Raspberry pi telegram bot and communicate with Raspberry Pi using Telegram App.

**Code:**

```

import time, datetime
import telepot
from telepot.loop import MessageLoop
now = datetime.datetime.now()
def action(msg):
 chat_id = msg['chat']['id']
 command = msg['text']
 print 'Received: %s' % command
 if command == '/hi':
 telegram_bot.sendMessage(chat_id, str("Hi! CircuitDigest"))
 elif command == '/time':
 telegram_bot.sendMessage(chat_id, str(now.hour)+str(":")+str(now.minute))
 elif command == '/logo':
 telegram_bot.sendPhoto(chat_id, photo = "https://i.pinimg.com/avatars/circuitdigest_1464122100_280.jpg")
 elif command == '/file':
 telegram_bot.sendDocument(chat_id, document=open('/home/pi/Aisha.py'))
 elif command == '/audio':
 telegram_bot.sendAudio(chat_id, audio=open('/home/pi/test.mp3'))
telegram_bot = telepot.Bot('468382312:AAFhURMxpVIMWEdFzbIQLszBPFEUpAeOLFQ')
print (telegram_bot.getMe())
MessageLoop(telegram_bot, action).run_as_thread()
print 'Up and Running....'
while 1:
 time.sleep(10)

```

## Project - 09

# Raspberry Pi Weather Station: Monitoring Humidity, Temperature and Pressure over Internet

---

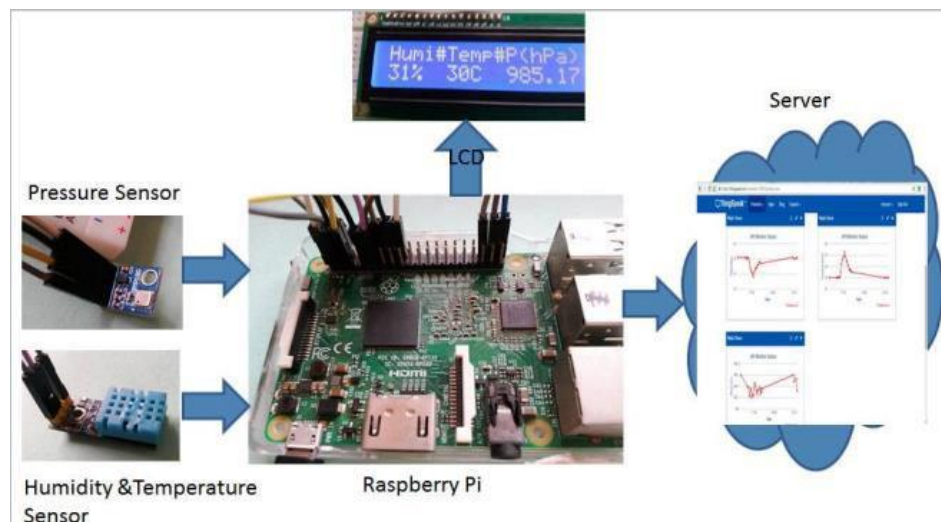
Humidity, Temperature and Pressure are three basic parameters to build any Weather Station and to measure environmental conditions. We have previously built a mini Weather Station using Arduino and this time we are extending the weather station with Raspberry Pi. This IoT based Project aims to show the current Humidity, Temperature and Pressure parameters on the LCD as well on the Internet server using Raspberry Pi, which makes it a Raspberry Pi Weather Station. You can install this setup anywhere and can monitor the weather conditions of that place from anywhere in the world over the internet, it will not only show the current data but can also show the past values in the form of Graphs.

We have used DHT11 Humidity & temperature sensor for sensing the temperature and BM180 Pressure sensor module for measuring barometric pressure. This Celsius scale Thermometer and percentage scale Humidity meter displays the ambient temperature and humidity through a LCD display and barometric pressure is displayed in millibar or hPa (hectopascal). All this data is sent to ThingSpeak server for live monitoring from anywhere in the world over internet. Do check the Demonstration Video and Python Program, given at the end of this tutorial.



## Working and ThingSpeak Setup:

This IoT based project has four sections. Firstly DHT11 sensor senses the Humidity & Temperature Data and BM180 sensor measures the atmospheric pressure. Secondly Raspberry Pi reads the DHT11 sensor module's output by using single wire protocol and BM180 pressure sensor's output by using I2C protocol and extracts both sensors values into a suitable number in percentage (humidity), Celsius scale (temperature), hectoPascal or millibar (pressure). Thirdly, these values are sent to ThingSpeak server by using inbuilt Wi-Fi of Raspberry Pi 3. And finally ThingSpeak analyses the data and shows it in a Graph form. A LCD is also used to display these values locally.



ThingSpeak provides very good tool for IoT based projects. By using ThingSpeak website, we can monitor our data and control our system over the Internet, using the Channels and webpages provided by ThingSpeak. ThingSpeak 'Collects' the data from the sensors, 'Analyze and Visualize' the data and 'Acts' by triggering a reaction. We have previously explained about sending data to ThingSpeak in detail, you can check there. Here we are briefly explaining to use ThingSpeak for this Raspberry Pi Weather station.

First you need to create account on ThingSpeak website and create a 'New channel' in it. In new channel you have to define some fields for the data you want to monitor, like in this project we will create three fields for Humidity, Temperature and Pressure data.

'Now click on 'API keys' tab and save the Write and Read API keys, here we are only using Write key. You need to Copy this key in 'key' variable in the Code.

## RPI Whether Station

Channel ID: **174207** | Whether Parameters Monitoring  
 Author: **saddam4201**  
 Access: **Public**

[Private View](#)
[Public View](#)
[Channel Settings](#)
[API Keys](#)
[Data Import / Export](#)


### Write API Key

Key **30BCDSRQ52AOI3UA**

[Generate New Write API Key](#)

After it, click on 'Data Import/Export' and copy the Update Channel Feed GET Request URL, which is:

[https://api.thingspeak.com/update?api\\_key=30BCDSRQ52AOI3UA&field1=0](https://api.thingspeak.com/update?api_key=30BCDSRQ52AOI3UA&field1=0)



[Channels](#)
[Apps](#)
[Blog](#)
[Support](#)

[Account](#)
[Sign Out](#)

## RPI Whether Station

Channel ID: **174207** | Whether Parameters Monitoring  
 Author: **saddam4201**  
 Access: **Public**

[Private View](#)
[Public View](#)
[Channel Settings](#)
[API Keys](#)
[Data Import / Export](#)

### Import

Upload a CSV file to import data into this channel

[Choose File](#) No file chosen

Time Zone **(GMT+05:30) New Delhi**

[Upload](#)

#### Update Channel Feed - GET

GET [https://api.thingspeak.com/update?api\\_key=30BCDSRQ52AOI3UA&field1=0](https://api.thingspeak.com/update?api_key=30BCDSRQ52AOI3UA&field1=0)

#### Update Channel Feed - POST

POST <https://api.thingspeak.com/update.json>  
 api\_key=30BCDSRQ52AOI3UA  
 field1=73

#### Get a Channel Feed

GET <https://api.thingspeak.com/channels/174207/feeds.json?results=2>

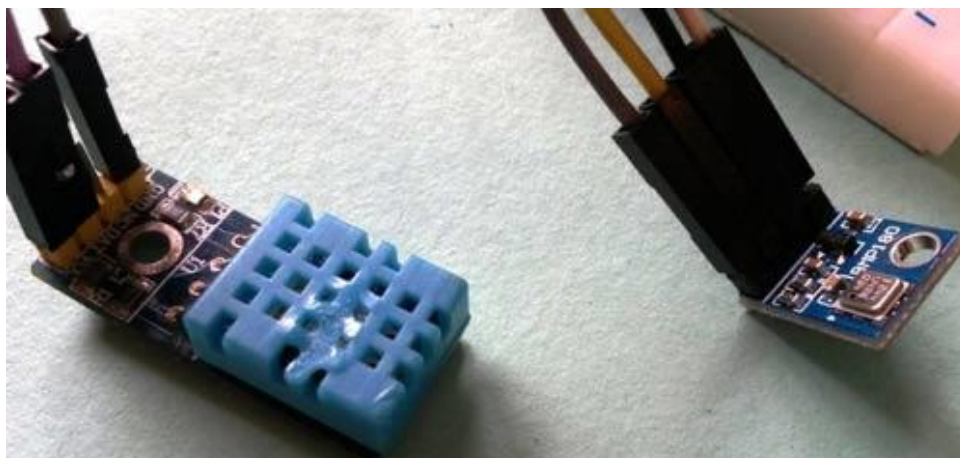
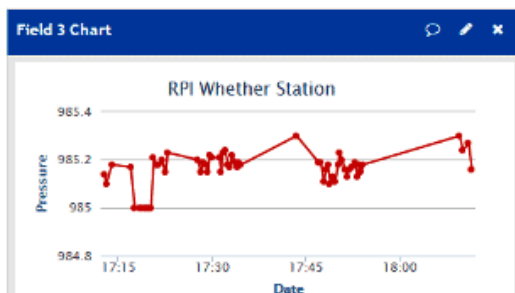
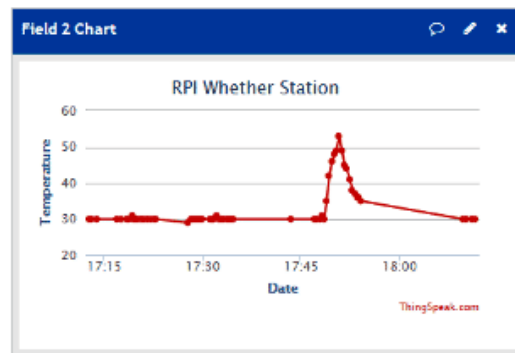
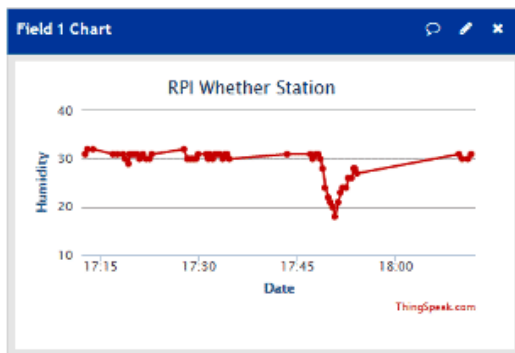
Now we need this 'Feed Get Request URL' in our Python code to open "api.thingspeak.com" and then send data using this Feed Request as query string. And Before sending data user needs to enter the temperature, humidity and pressure data in this query String using variables in program, check in the Code at the end this article.



```
URL = 'https://api.thingspeak.com/update?api_key=%s' % key
```

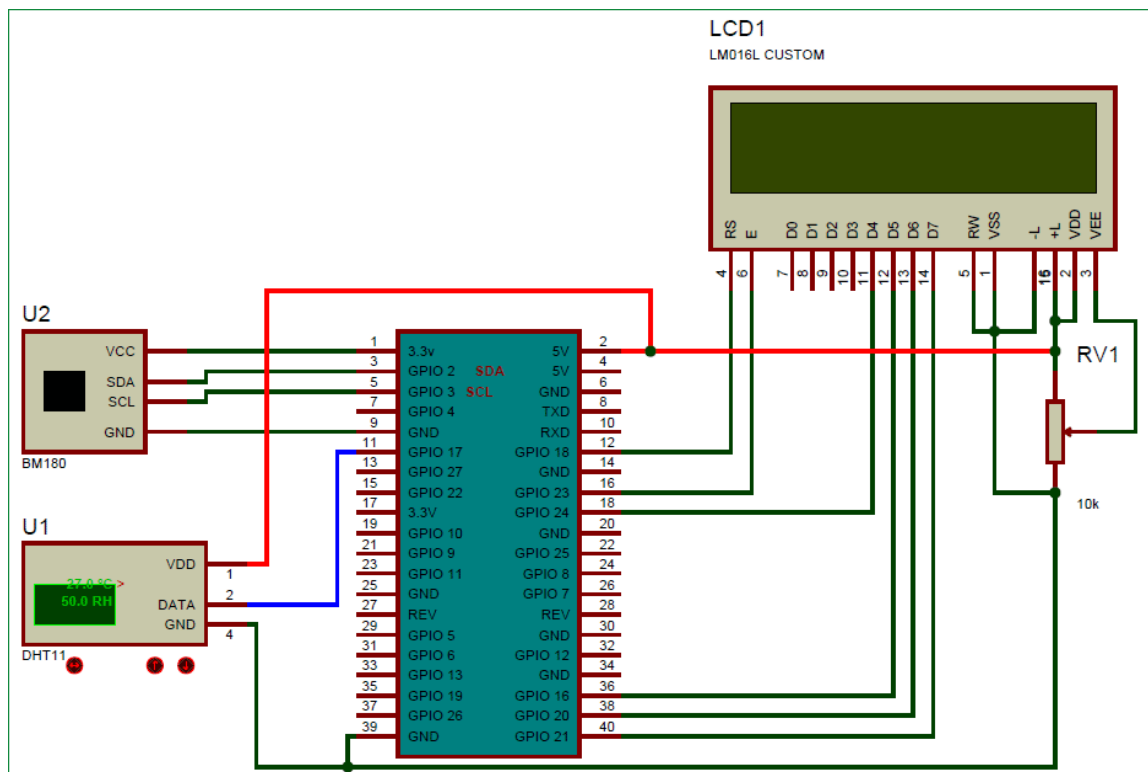
```
finalURL = URL + "&field1=%s&field2=%s"%(humi, temp)+"&field3=%s" %(pressure)
```

Working of DHT11 is based on single wire serial communication for fetching data from DHT11. Here we have used AdaFruit DHT11 library for interfacing DHT11 with Raspberry Pi. Raspberry Pi here collects the Humidity and temperature data from DHT11 and atmospheric pressure from BMP180 sensor and then sends it to 16x2 LCD and ThingSpeak server. ThingSpeak displays the Data in form of Graph as below:



You can learn more about DHT11 Sensor and its Interfacing with Arduino [here](#).

## Circuit Diagram:



## Raspberry Pi Configuration and Python Program:

We are using Python language here for the Program. Before coding, user needs to configure Raspberry Pi. You can check our previous tutorials for Getting Started with Raspberry Pi and Installing & Configuring Raspbian Jessie OS in Pi.

First off all we need to install Adafruit Python DHT Sensor Library files to run this project on Raspberry Pi. To do this we need to follow given commands:

```
sudo apt-get install git-core

sudo apt-get update

git clone https://github.com/adafruit/Adafruit_Python_DHT.git

cd Adafruit_Python_DHT

sudo apt-get install build-essential python-dev

sudo python setup.py install
```

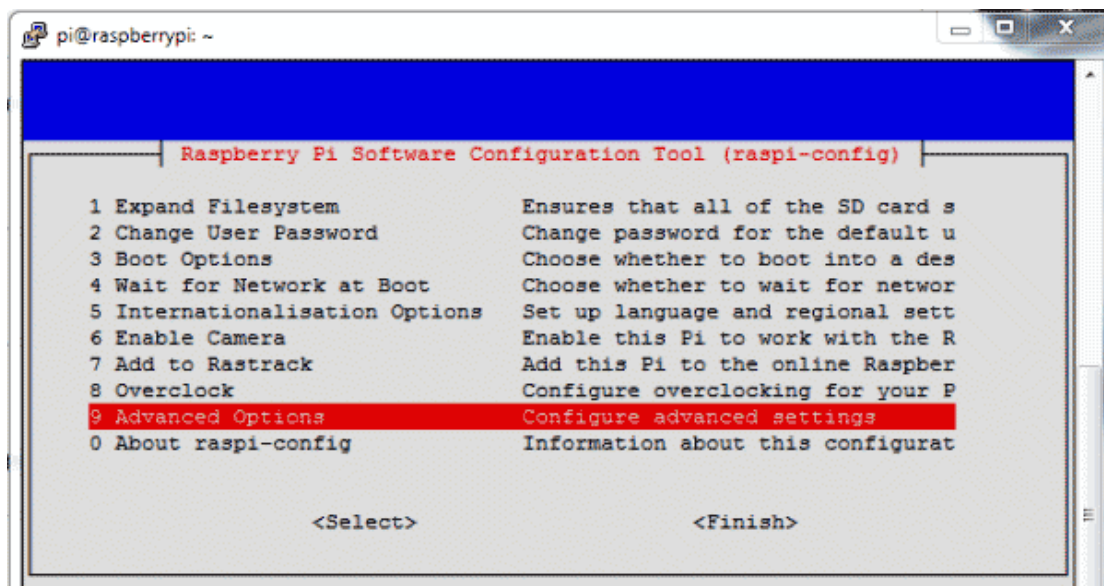


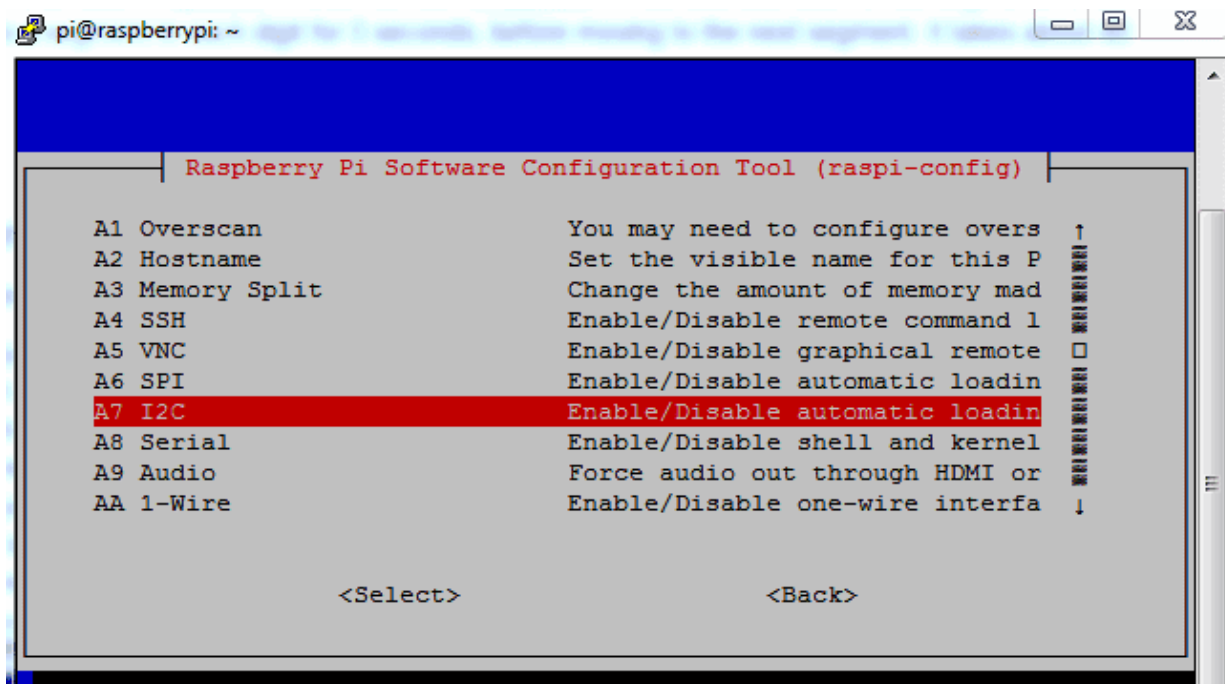
```
pi@raspberrypi: ~/Adafruit_Python_DHT
pi@raspberrypi:~ $ sudo apt-get install git-core
Reading package lists... Done
Building dependency tree
Reading state information... Done
git-core is already the newest version.
The following packages were automatically installed and are no longer required:
 libasn1-8-heimdal libgssapi3-heimdal libhcrypto4-heimdal
 libheimbase1-heimdal libheimntlm0-heimdal libhx509-5-heimdal
 libkrb5-26-heimdal libroken18-heimdal libwind0-heimdal libxfce4ui-1-0
 pypy-upstream-doc xfce-keyboard-shortcuts
Use 'apt-get autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.
pi@raspberrypi:~ $ git clone https://github.com/adafruit/Adafruit_Python_DHT.git
fatal: destination path 'Adafruit_Python_DHT' already exists and is not an empty directory.
pi@raspberrypi:~ $ cd Adafruit_Python_DHT
pi@raspberrypi:~/Adafruit_Python_DHT $ sudo apt-get install build-essential python-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
build-essential is already the newest version.
```

After this, user needs to enable Raspberry Pi I2C by going into RPi Software Configuration Tool:

```
sudo raspi-config
```

Then go to 'Advance Options', select 'I2C' and 'Enable' it.





Programming part of this project plays a very important role to perform all the operations. First of all we include all required libraries, initiaze variables and define pins for LCD and DHT11.

```
import sys

import RPi.GPIO as GPIO

import os

import Adafruit_DHT

import urllib2

import smbus

import time

from ctypes import c_short

#Register Address

regCall = 0xAA

... ..

.....
```

In def main(): function, below code is used for sending the data to the server and display it over the LCD, continuously in while loop.

```
def main():
 print 'System Ready...'

 URL = 'https://api.thingspeak.com/update?api_key=%s' % key

 print "Wait...."

 while True:
 (humi, temp)= readDHT()
 (pressure) =readBmp180()

 lcdcmd(0x01)
 lcdstring("Humi#Temp#P(hPa)")
 lcdstring(humi+'%'+" %sC %s" %(temp, pressure))
 finalURL = URL + "&field1=%s&field2=%s"%(humi, temp)+"&field3=%s" %(pressure)
 print finalURL
 s=urllib2.urlopen(finalURL);
 print humi+ " " + temp + " " + pressure
 s.close()
 time.sleep(10)
```

For LCD, def lcd\_init() function is used to initialize LCD in four bit mode, def lcdcmd(ch) function is used for sending command to LCD, def lcddata(ch) function is used for sending data to LCD and def lcdstring(Str) function is used to send data string to LCD. You can check all these functions in Code given afterwards.

Given def readDHT() function is used for reading DHT11 Sensor:

```
def readDHT():
```

```

humi, temp = Adafruit_DHT.read_retry(Adafruit_DHT.DHT11, DHTpin)

return (str(int(humi)), str(int(temp)))

```

def readBmp180 function is used for reading pressure from the BM180 sensor. BM180 sensor can also give temperature but here we have only used it for calculating pressure.

```

def readBmp180(addr=deviceAdd):

 value = bus.read_i2c_block_data(addr, regCall, 22) # Read calibration data

 # Convert byte data to word values

 AC1 = convert1(value, 0)

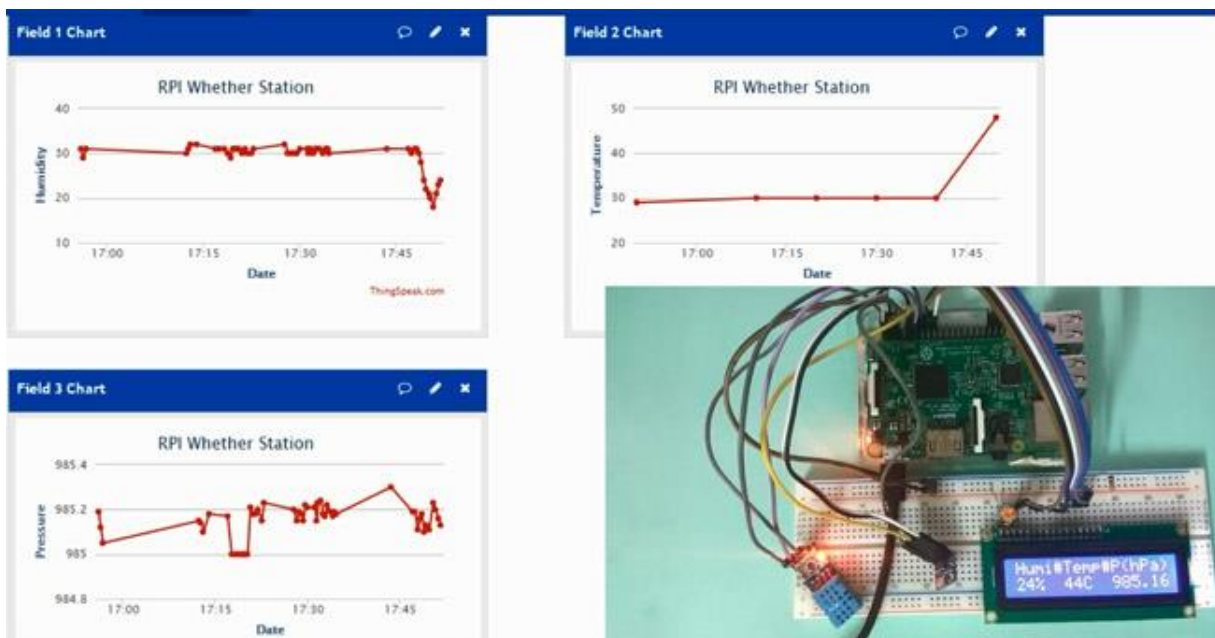
 AC2 = convert1(value, 2)

 AC3 = convert1(value, 4)

 AC4 = convert2(value, 6)


```

So this is the basic Raspberry Pi Weather Station, you can further extend it to measure various weather related parameters like wind speed, soil temperature, illuminance (lux), rainfall, air quality etc.



## Code:

```
import sys
import RPi.GPIO as GPIO
import os
import Adafruit_DHT
import urllib2
import smbus
import time
from ctypes import c_short
#Register Address
regCall = 0xAA
regMean = 0xF4
regMSB = 0xF6
regLSB = 0xF7
regPres = 0x34
regTemp = 0x2e
DEBUG = 1
sample = 2
deviceAdd = 0x77
```

```

humi=""
temp=""
#bus = smbus.SMBus(0) #for Pi1 uses 0
I2cbus = smbus.SMBus(1) # for Pi2 uses 1
DHTpin = 17
key="30BCDSRQ52AOI3UA" # Enter your Write API key from ThingSpeak
GPIO.setmode(GPIO.BCM)
Define GPIO to LCD mapping
LCD_RS = 18
LCD_EN = 23
LCD_D4 = 24
LCD_D5 = 16
LCD_D6 = 20
LCD_D7 = 21
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(LCD_E, GPIO.OUT)
GPIO.setup(LCD_RS, GPIO.OUT)
GPIO.setup(LCD_D4, GPIO.OUT)
GPIO.setup(LCD_D5, GPIO.OUT)
GPIO.setup(LCD_D6, GPIO.OUT)
GPIO.setup(LCD_D7, GPIO.OUT)
def convert1(data, i): # signed 16-bit value
 return c_short((data[i]<< 8) + data[i + 1]).value
def convert2(data, i): # unsigned 16-bit value
 return (data[i]<< 8) + data[i+1]
def readBmp180(addr=deviceAdd):
 value = bus.read_i2c_block_data(addr, regCall, 22) # Read calibration data
 # Convert byte data to word values
 AC1 = convert1(value, 0)
 AC2 = convert1(value, 2)
 AC3 = convert1(value, 4)

```

```

AC4 = convert2(value, 6)
AC5 = convert2(value, 8)
AC6 = convert2(value, 10)
B1 = convert1(value, 12)
B2 = convert1(value, 14)
MB = convert1(value, 16)
MC = convert1(value, 18)
MD = convert1(value, 20)

Read temperature
bus.write_byte_data(addr, regMean, regTemp)
time.sleep(0.005)
(msb, lsb) = bus.read_i2c_block_data(addr, regMSB, 2)
P2 = (msb << 8) + lsb

Read pressure
bus.write_byte_data(addr, regMean, regPres + (sample << 6))
time.sleep(0.05)
(msb, lsb, xsb) = bus.read_i2c_block_data(addr, regMSB, 3)
P1 = ((msb << 16) + (lsb << 8) + xsb) >> (8 - sample)

Refine temperature
X1 = ((P2 - AC6) * AC5) >> 15
X2 = (MC << 11) / (X1 + MD)
B5 = X1 + X2
temperature = (B5 + 8) >> 4

Refine pressure
B6 = B5 - 4000
B62 = B6 * B6 >> 12
X1 = (B2 * B62) >> 11
X2 = AC2 * B6 >> 11
X3 = X1 + X2
B3 = (((AC1 * 4 + X3) << sample) + 2) >> 2
X1 = AC3 * B6 >> 13
X2 = (B1 * B62) >> 16

```

```

X3 = ((X1 + X2) + 2) >> 2
B4 = (AC4 * (X3 + 32768)) >> 15
B7 = (P1 - B3) * (50000 >> sample)
P = (B7 * 2) / B4
X1 = (P >> 8) * (P >> 8)
X1 = (X1 * 3038) >> 16
X2 = (-7357 * P) >> 16
pressure = P + ((X1 + X2 + 3791) >> 4)
return (str(pressure/100.0))

def readDHT():
 humi, temp = Adafruit_DHT.read_retry(Adafruit_DHT.DHT11, DHTpin)
 return (str(int(humi)), str(int(temp)))

def lcd_init():
 lcdcmd(0x33)
 lcdcmd(0x32)
 lcdcmd(0x06)
 lcdcmd(0x0C)
 lcdcmd(0x28)
 lcdcmd(0x01)
 time.sleep(0.0005)

def lcdcmd(ch):
 GPIO.output(RS, 0)
 GPIO.output(D4, 0)
 GPIO.output(D5, 0)
 GPIO.output(D6, 0)
 GPIO.output(D7, 0)
 if ch&0x10==0x10:
 GPIO.output(D4, 1)
 if ch&0x20==0x20:
 GPIO.output(D5, 1)
 if ch&0x40==0x40:
 GPIO.output(D6, 1)

```



```

if ch&0x80==0x80:
 GPIO.output(D7, 1)
GPIO.output(EN, 1)
time.sleep(0.0005)
GPIO.output(EN, 0)
Low bits
GPIO.output(D4, 0)
GPIO.output(D5, 0)
GPIO.output(D6, 0)
GPIO.output(D7, 0)
if ch&0x01==0x01:
 GPIO.output(LCD_D4, 1)
if ch&0x02==0x02:
 GPIO.output(LCD_D5, 1)
if ch&0x04==0x04:
 GPIO.output(LCD_D6, 1)
if ch&0x08==0x08:
 GPIO.output(LCD_D7, 1)
GPIO.output(EN, 1)
time.sleep(0.0005)
GPIO.output(EN, 0)
def lcddata(ch):
 GPIO.output(RS, 1)
 GPIO.output(D4, 0)
 GPIO.output(D5, 0)
 GPIO.output(D6, 0)
 GPIO.output(D7, 0)
 if ch&0x10==0x10:
 GPIO.output(D4, 1)
 if ch&0x20==0x20:
 GPIO.output(D5, 1)
 if ch&0x40==0x40:

```

```

 GPIO.output(D6, 1)
if ch&0x80==0x80:
 GPIO.output(D7, 1)
GPIO.output(EN, 1)
time.sleep(0.0005)
GPIO.output(EN, 0)
Low bits
GPIO.output(D4, 0)
GPIO.output(D5, 0)
GPIO.output(D6, 0)
GPIO.output(D7, 0)
if ch&0x01==0x01:
 GPIO.output(LCD_D4, 1)
if ch&0x02==0x02:
 GPIO.output(LCD_D5, 1)
if ch&0x04==0x04:
 GPIO.output(LCD_D6, 1)
if ch&0x08==0x08:
 GPIO.output(LCD_D7, 1)
GPIO.output(EN, 1)
time.sleep(0.0005)
GPIO.output(EN, 0)
def lcdstring(Str):
 l=0;
 l=len(Str)
 for i in range(l):
 lcddata(ord(message[i]))
lcd_init()
lcdcmd(0x01)
lcdstring("Circuit Digest")
lcdcmd(0xc0)
lcdstring("Welcomes you")

```

```

time.sleep(3) # 3 second delay
main() function
def main():
 print 'System Ready...'
 URL = 'https://api.thingspeak.com/update?api_key=%s' % key
 print "Wait...."
 while True:
 (humi, temp)= readDHT()
 (pressure) =readBmp180()
 lcdcmd(0x01)
 lcdstring("Humi#Temp#P(hPa)")
 lcdstring(humi+'%'+" %sC %s" %(temp, pressure))
 finalURL = URL + "&field1=%s&field2=%s"%(humi, temp)+"&field3=%s" %(pressure)
 print finalURL
 s=urllib2.urlopen(finalURL);
 print humi+ " " + temp + " " + pressure
 s.close()
 time.sleep(10)
if __name__=="__main__":
 main()

```









**Imran Khan Patan**, working as Assistant Professor, Department of Electronics & Communication Engineering, St.Johns College of Engineering & Technology (Autonomous), Yemmiganur-518360, Kurnool(D), A.P. Currently he is doing his PhD Research in the field of Microwave Communications at JNTUA, Anantapur. He has published 34 Quality papers in a variety of Journals and at National and International Conferences. Additionally, he has provided his skills as a Reviewer to different IEEE, Springer & Scopus Conferences. He also received Best Reviewer Award for his Contribution / Commitment in 9th ICCM-2023, NERIST-Arunachal Pradesh & California State University-USA & France International Association of Academicians (IAASSE), USA.



**Mahaboob Basha S**, orking as Assistant Professor, Department of Electronics & Communication Engineering, St.Johns College of Engineering & Technology (Autonomous), Yemmiganur-518360, Kurnool(D), A.P. He is a Professional Life member for ISTE & IETE Organizations. He is knowledgeable in wide range of subjects. He has Published 21 Quality Papers in various Journals, National/International Conferences. He also provided his skills as a Reviewer to SASI-ITE'24 conference.



**Ahmed Basha Syed**, working as Assistant Professor, Department of Electronics & Communication Engineering, St.Johns College of Engineering & Technology (Autonomous), Yemmiganur-518360, Kurnool(D), A.P. He is knowledgeable in wide range of subjects. He has Published 21 Quality Papers in various Journals, National/International Conferences. He is a Professional Life member for IEI & IAENG Organizations.



**AMARAVATHI  
RESEARCH ACADEMY**  
*Communicating your research*



amazon  
kindle



Google Play

Google books



ISBN : 978-93-6285-228-1



978-93-6285-228-1

Rs.450/-

**ADDRESS:**

Amaravathi Research  
Academy, Plot no. 15,  
Royal Garden, Kompally  
Village, MD, Medchal,  
Hyderabad, Telangana  
500014

Date of Publish : 09-09-2024